Laboratório de
Tecnologia da Informação
Aplicada

Controle de Versão usando o Git

Marcelo Augusto Cordeiro

Gabriel Luiz Bastos de Oliveira

Luís Henrique Puhl de Souza

marcelo.augusto.cordeiro@gmail.com

gabiluiz@gmail.com

luispuhl@gmail.com

Laboratório de
Tecnologia da Informação
Aplicada

LTIA

CENTRO DE PESQUISA
CATI
SEPIN/MCT

Agenda

- O que é e por que usar Controle de Versão
- Por que usar o Git
- Configurações
- Comandos Básicos
- Branching
- Remote Branches
- Desfazendo Mudanças
- GitHub

O que é Controle de Versão

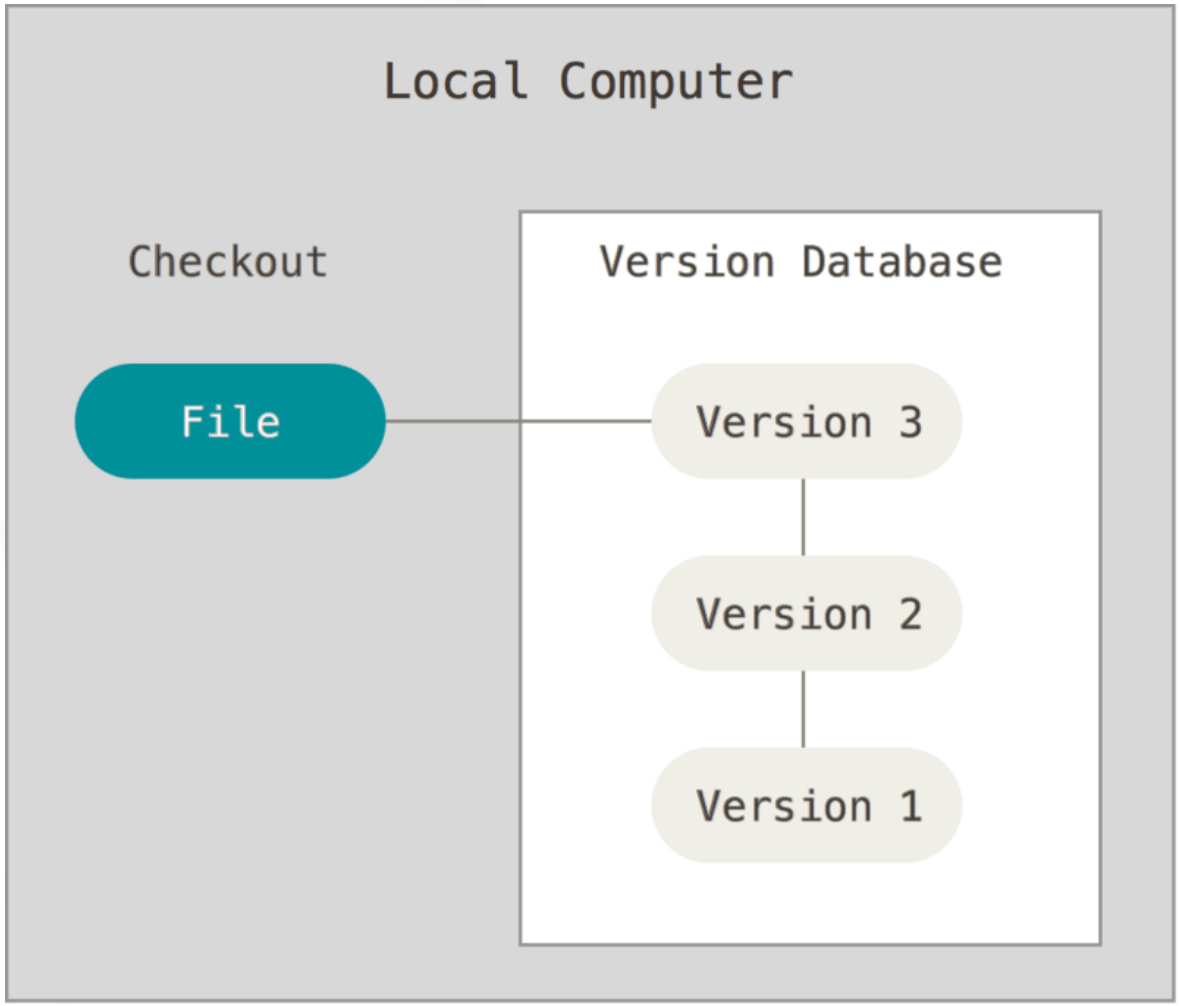
O controle de versão é um sistema que registra as mudanças feitas em um arquivo ou em um conjunto de arquivos ao longo do tempo de forma que você possa recuperar versões específicas.

Pro Git, Scott Chacon e Ben Straub

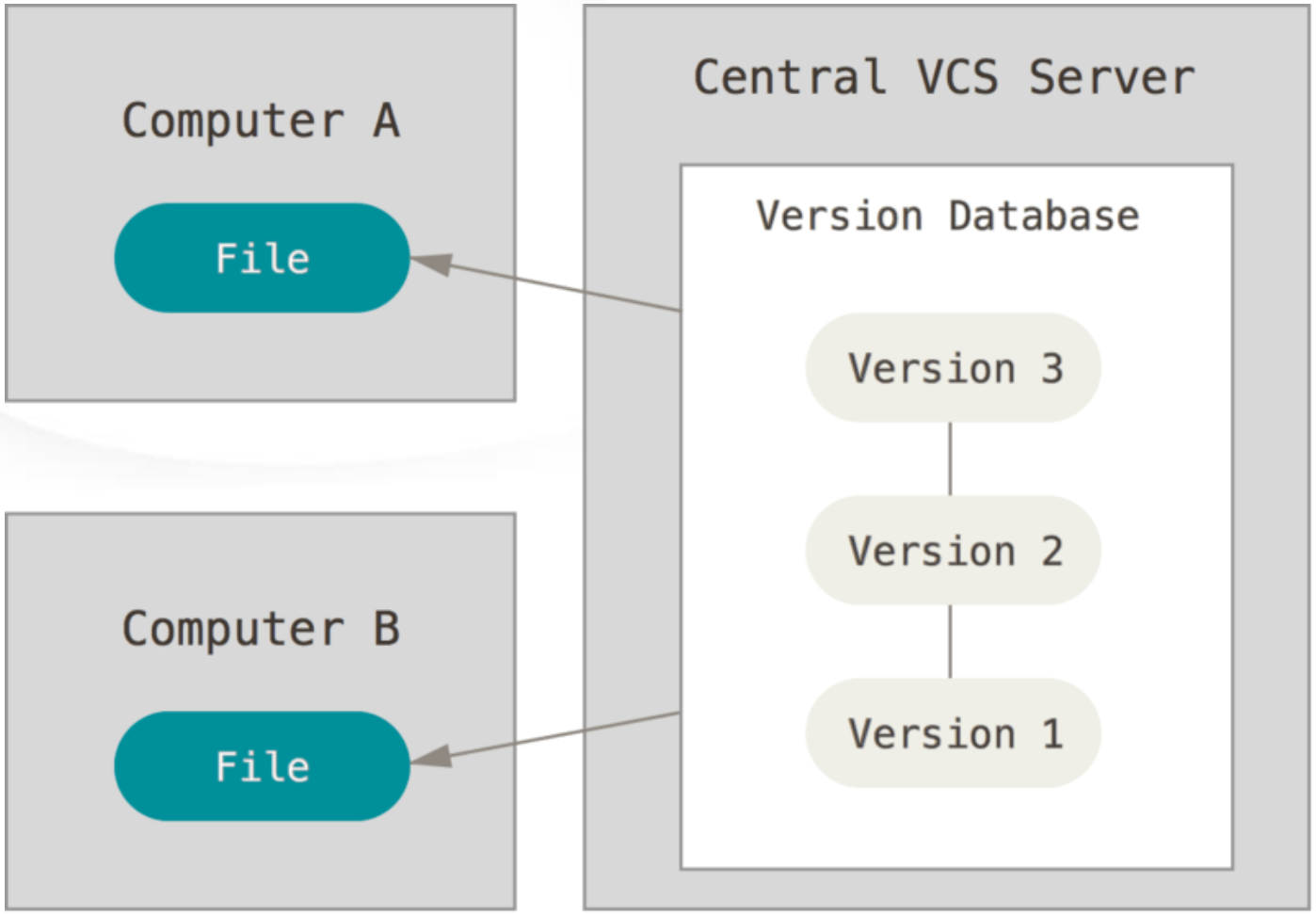
Por que usar Controle de Versão

- Compartilhamento e versionamento de qualquer arquivo;
- Trabalho em paralelo com fácil controle das modificações;
- Possibilidade de trabalhar em “equipes” enormes (centenas, até milhares de pessoas).

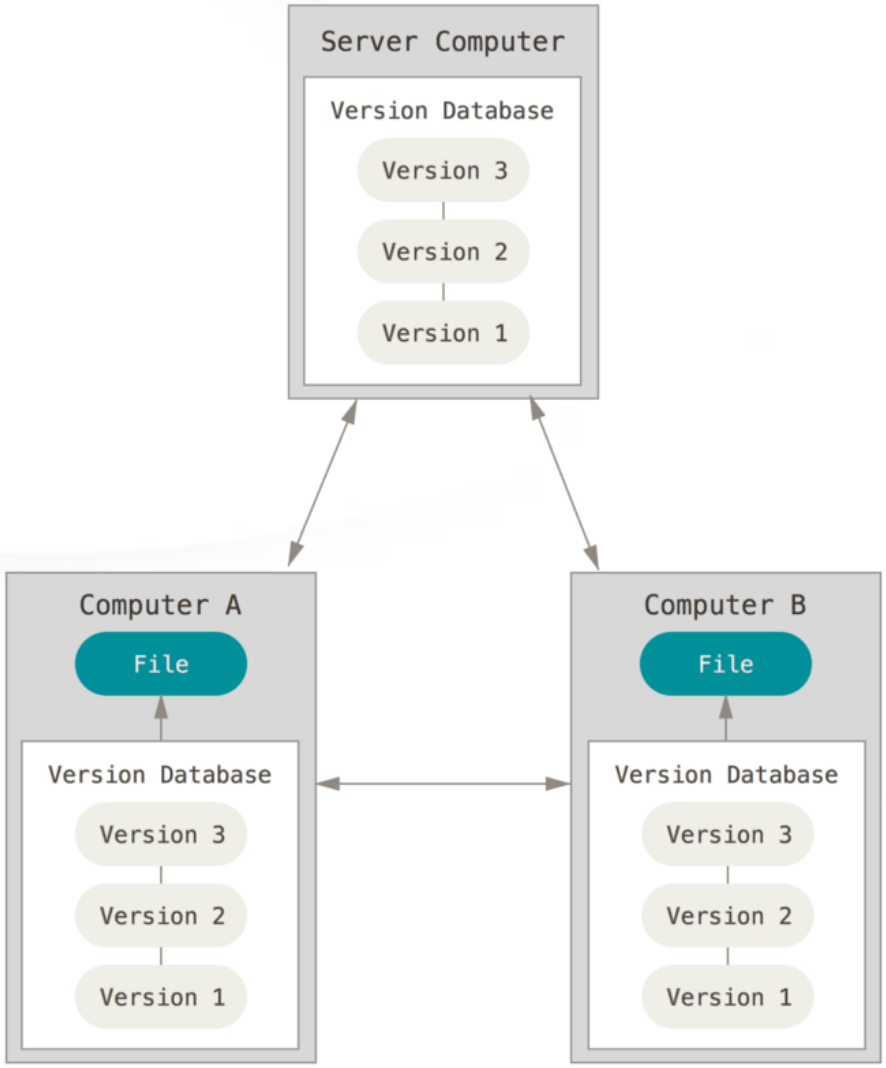
Controle de Versão Local



Controle de Versão Centralizado



Controle de Versão Distribuído



Por que usar o Git?

- Implementação de Branches extremamente leve;
- Comunidade OpenSource do GitHub.

Configurando o Git

- git config --global user.name “Nome Sobrenome"
- git config --global user.email email@gmail.com
- git config --global core.editor “notepad"

- git config --list

- git help comando

Criando um Repositório

- git init

- git clone https://github.com/user/Project.git

Comandos Básicos

Crie um diretório git

Working Dir.	Staging Area	.git Dir.

Comandos Básicos

Crie um arquivo qualquer .txt

Working Dir.

Staging Area

.git Dir.

Comandos Básicos

git status

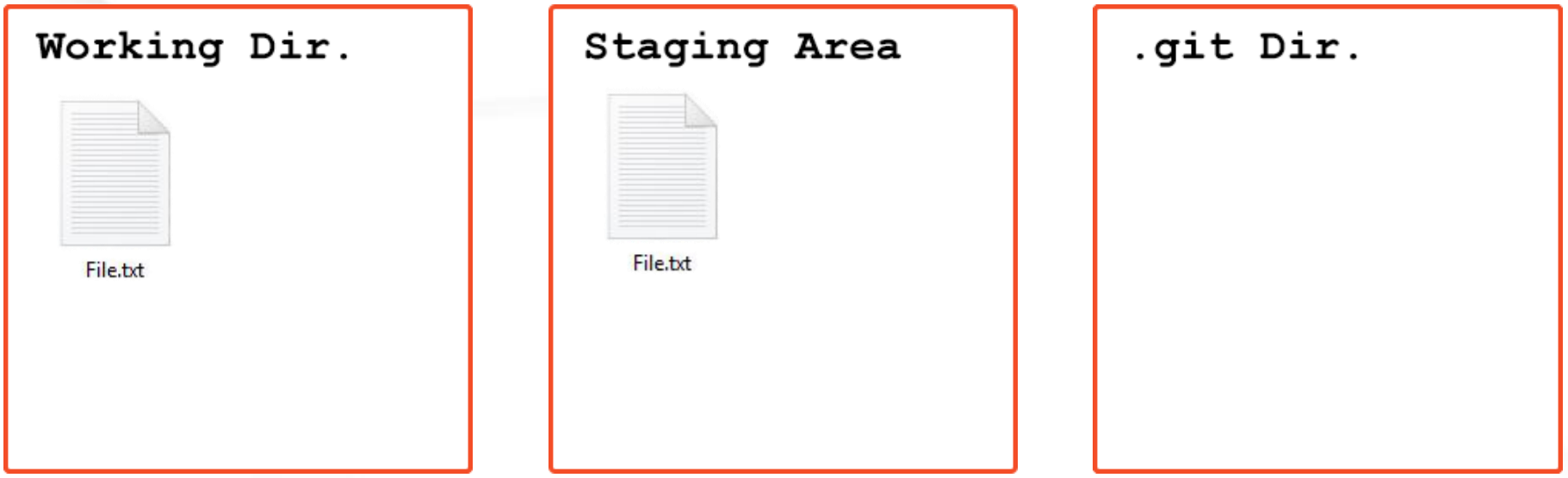
Working Dir.

Staging Area

.git Dir.

Comandos Básicos

git add File.txt

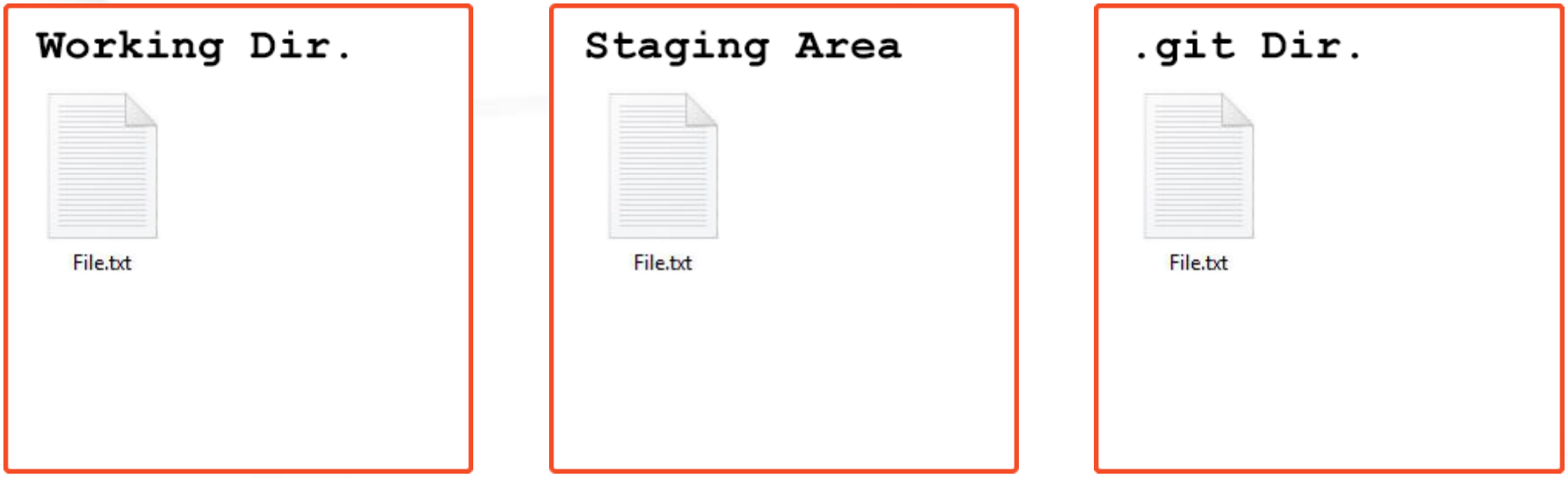


Comandos Básicos

git status

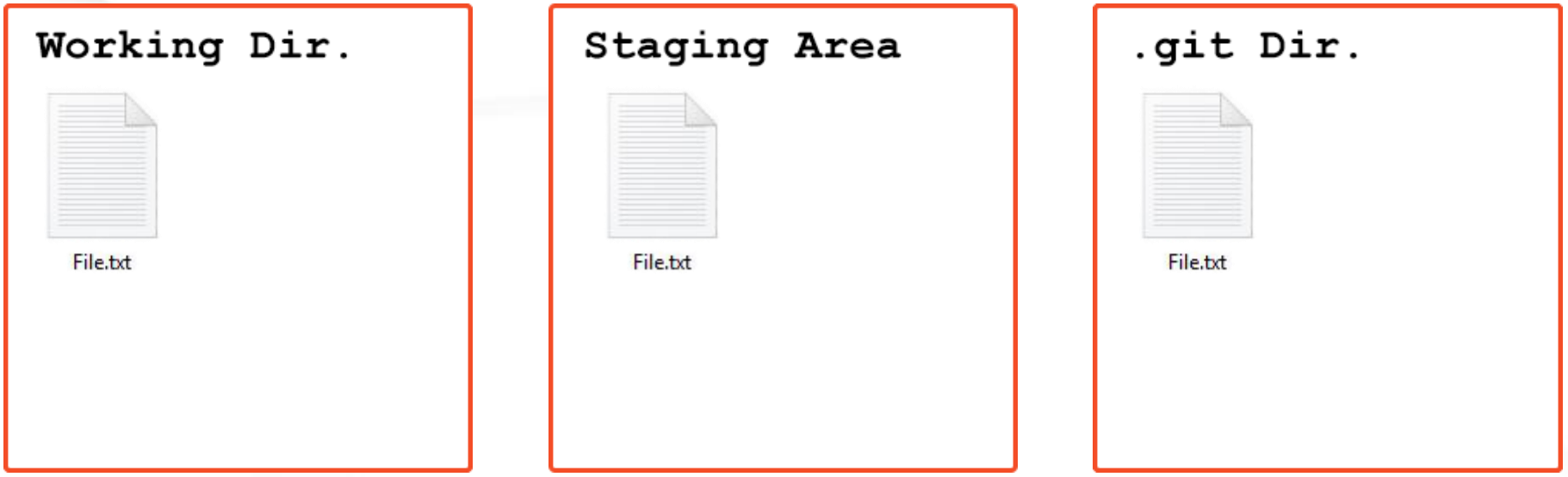
Comandos Básicos

```
git commit –m “Adicionei File.txt”
```



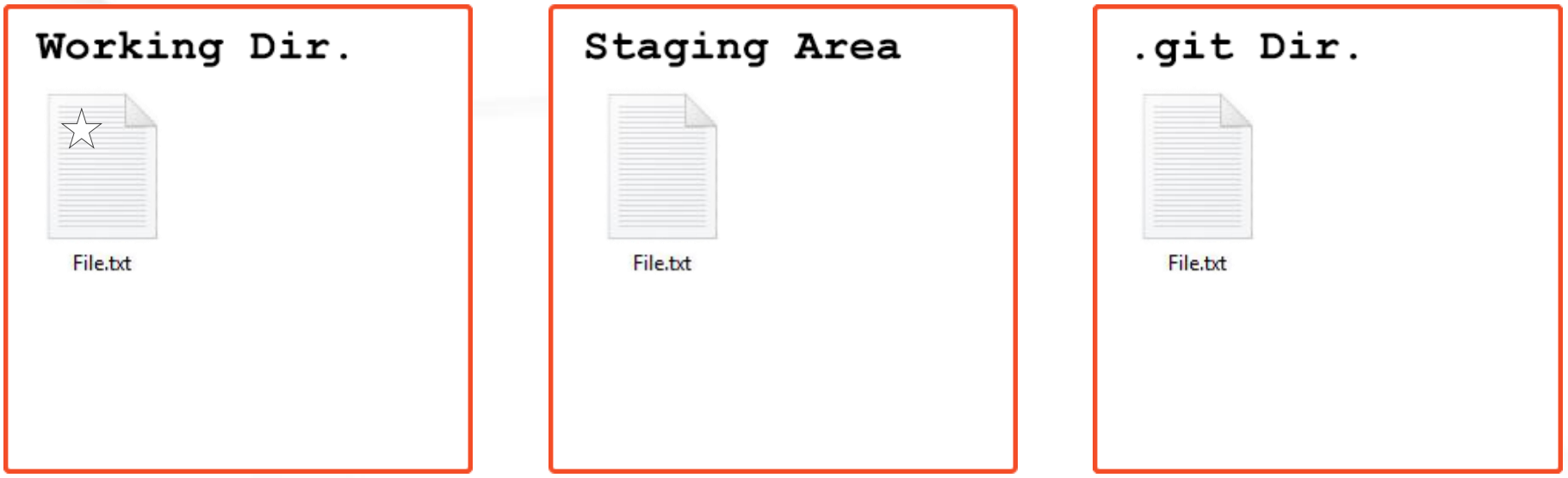
Comandos Básicos

git status



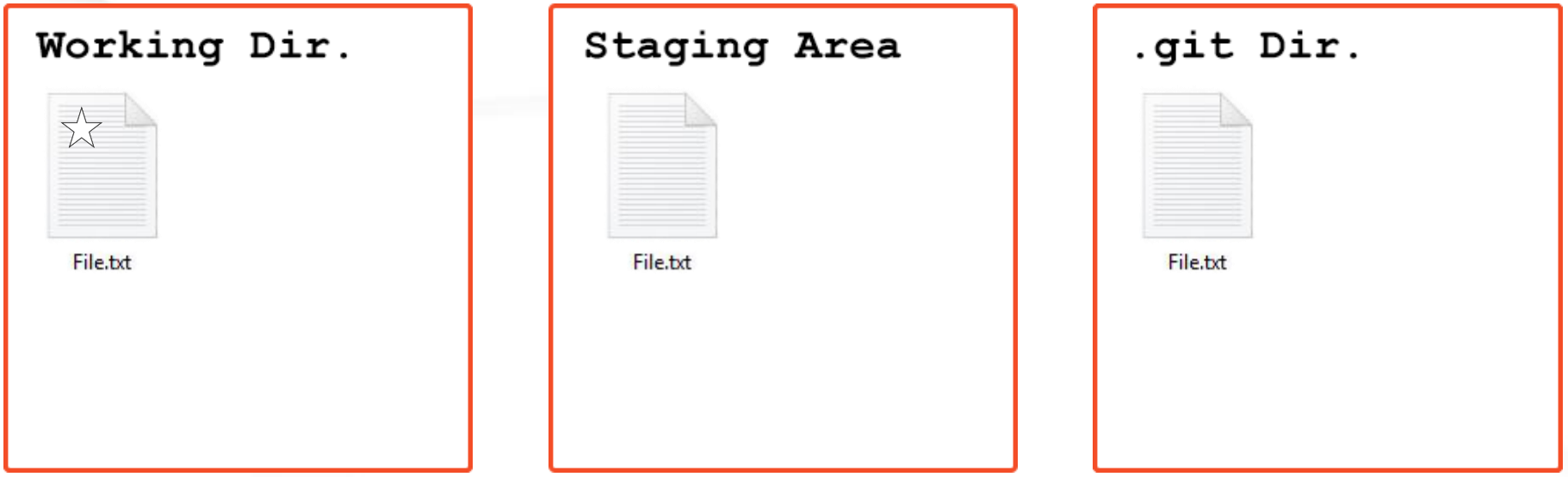
Comandos Básicos

Edite o arquivo File.txt



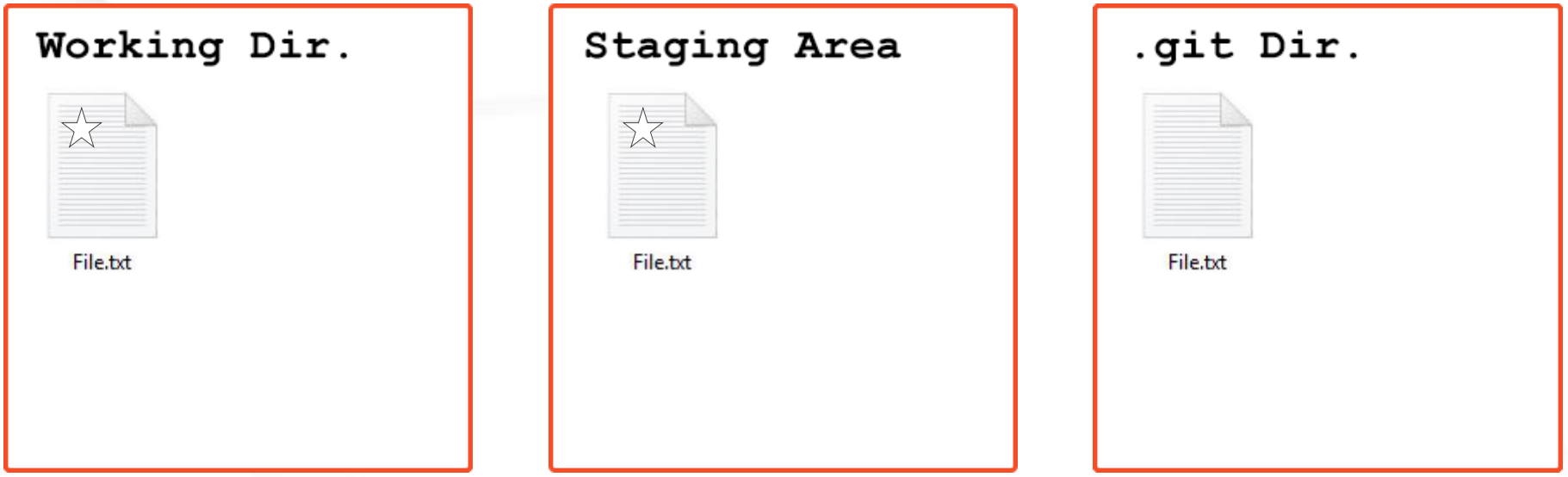
Comandos Básicos

git status



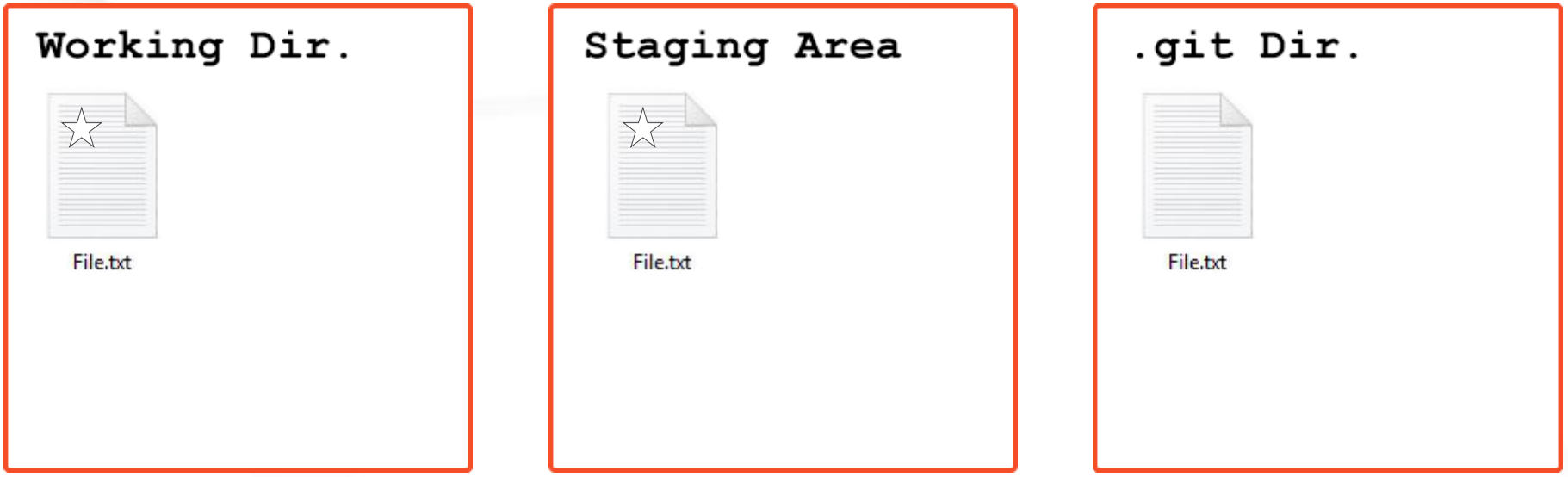
Comandos Básicos

git add *



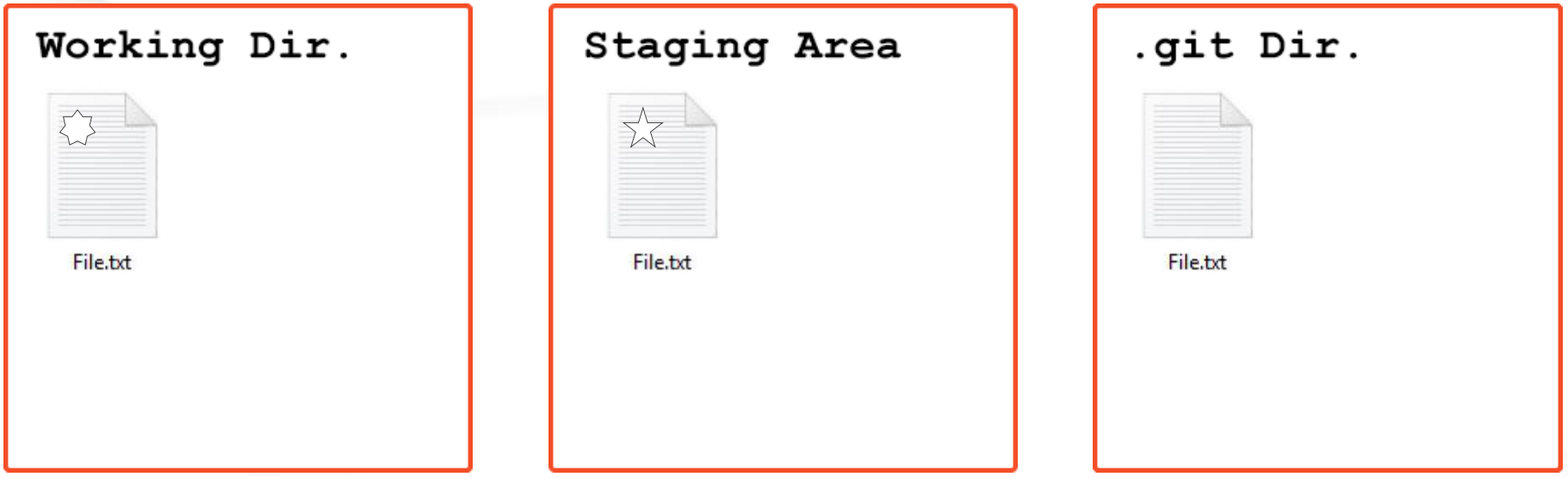
Comandos Básicos

git status



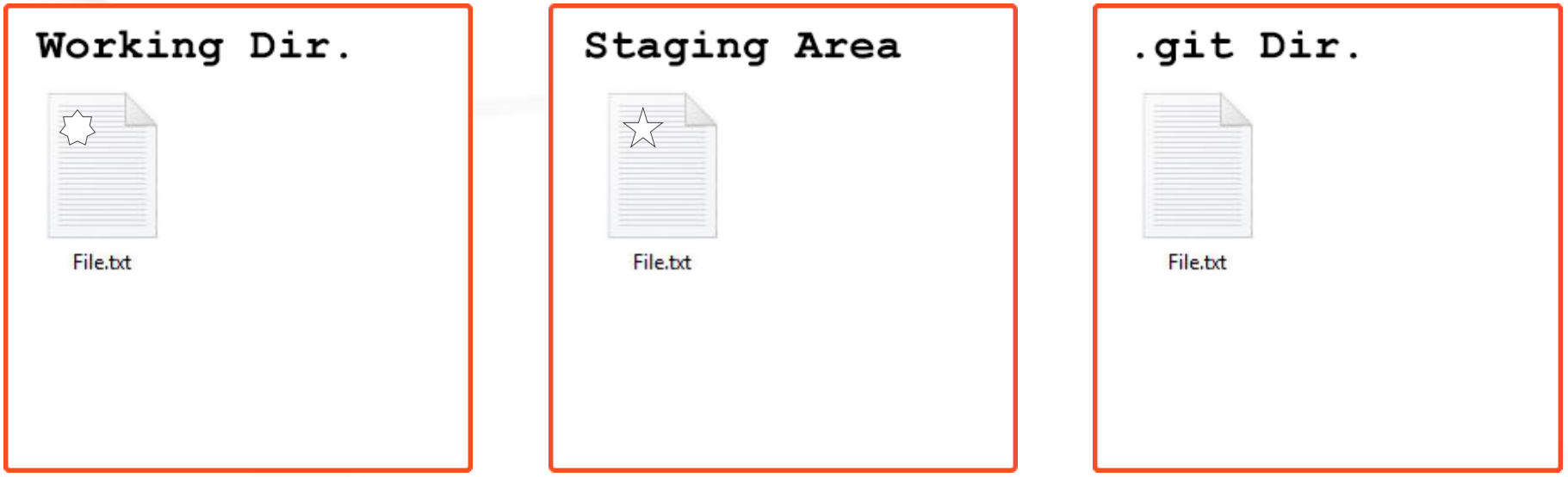
Comandos Básicos

Edite novamente File.txt



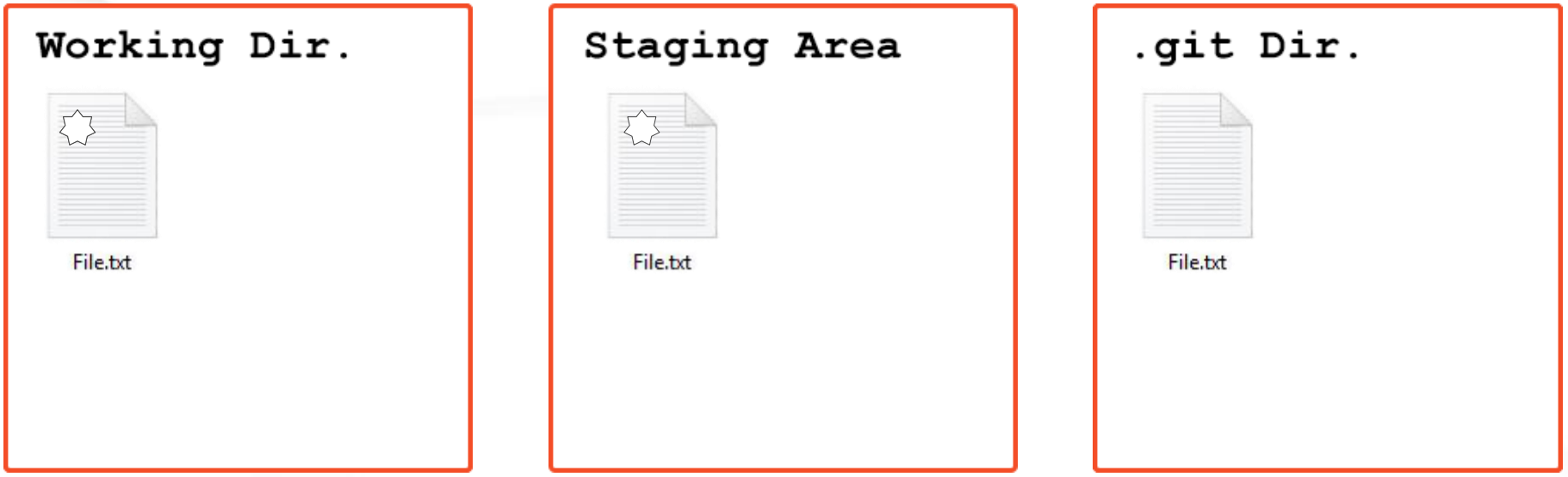
Comandos Básicos

git status



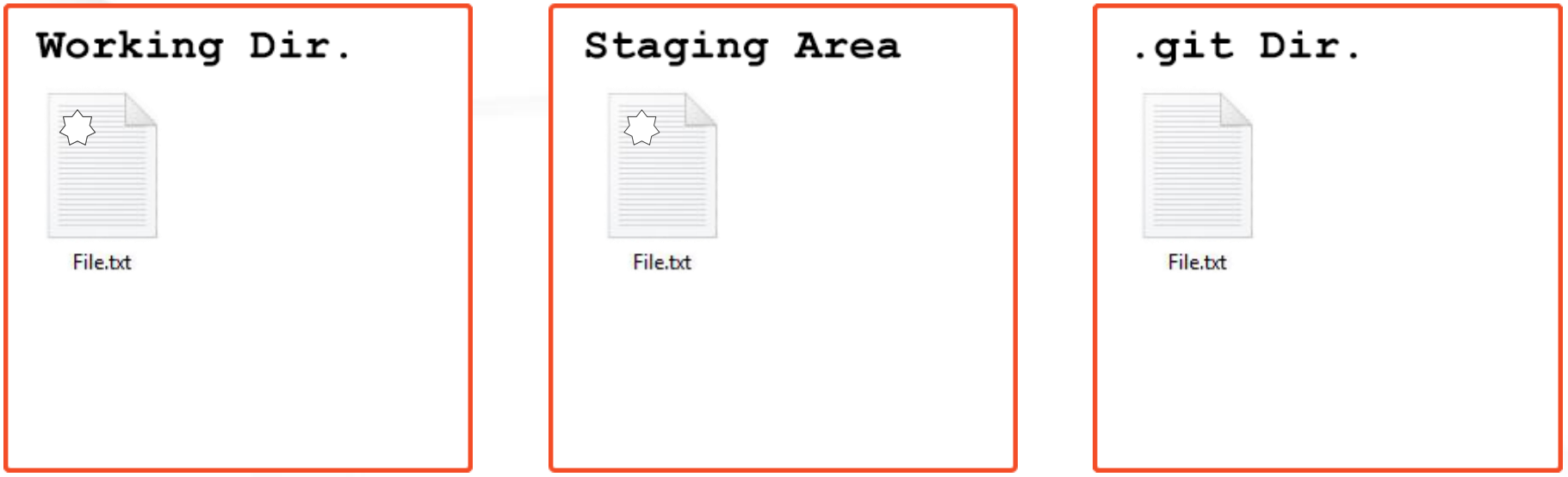
Comandos Básicos

```
git add *.txt
```



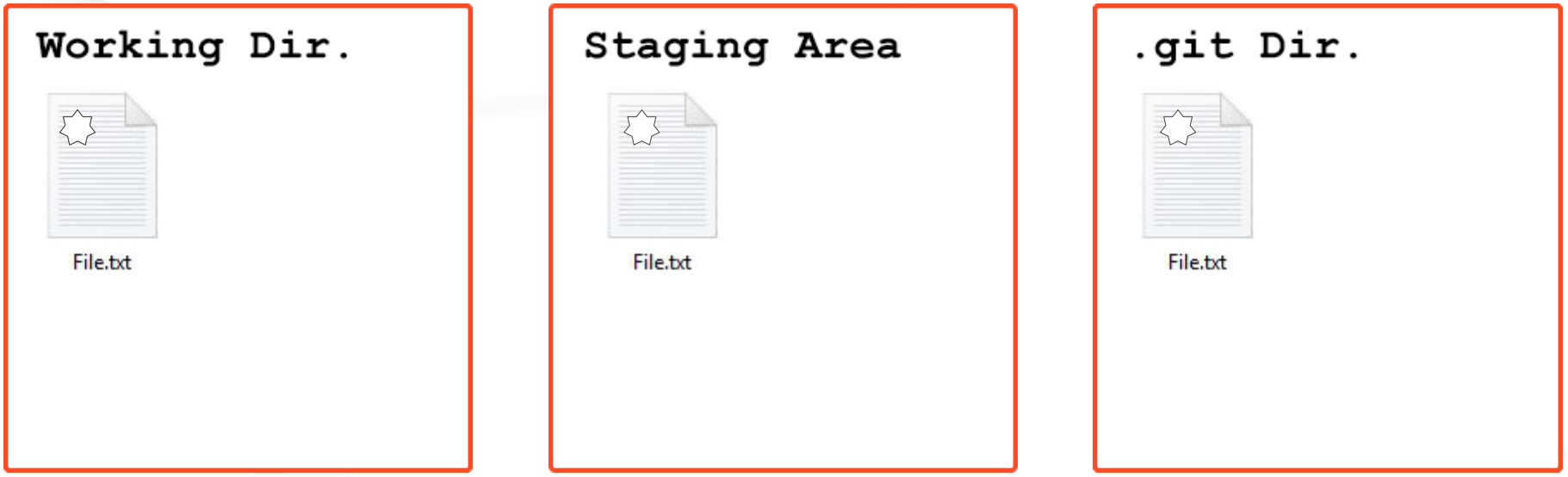
Comandos Básicos

git status



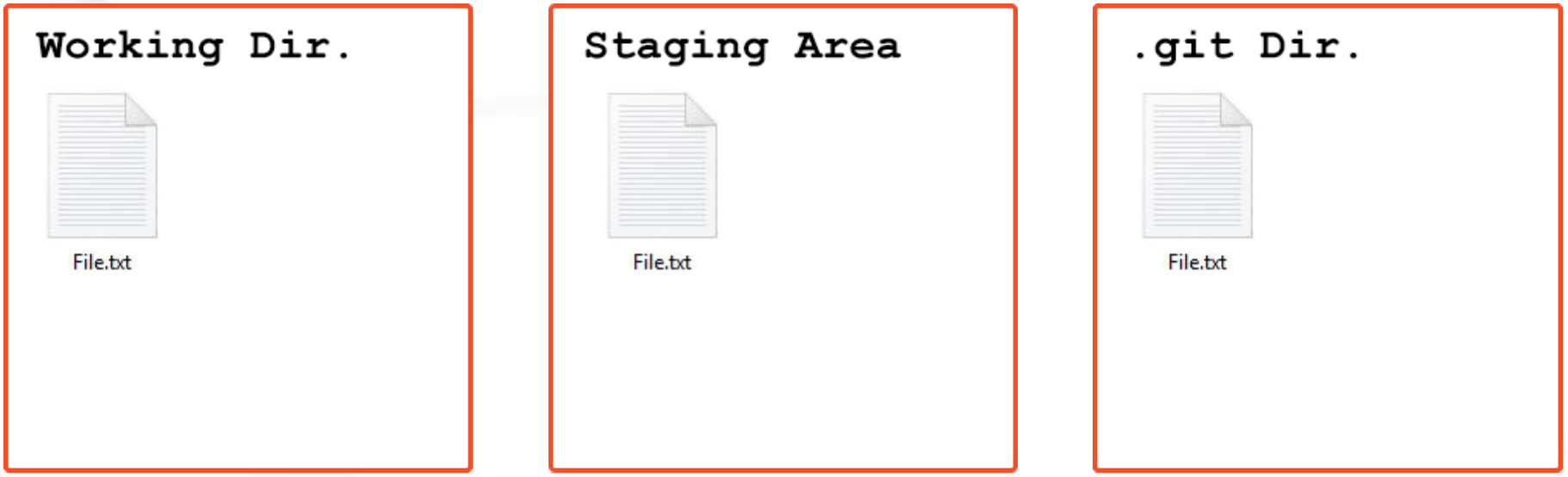
Comandos Básicos

```
git commit -m “Alterei File.txt”
```



Comandos Básicos

git log



Comandos Básicos

Delete o arquivo File.txt

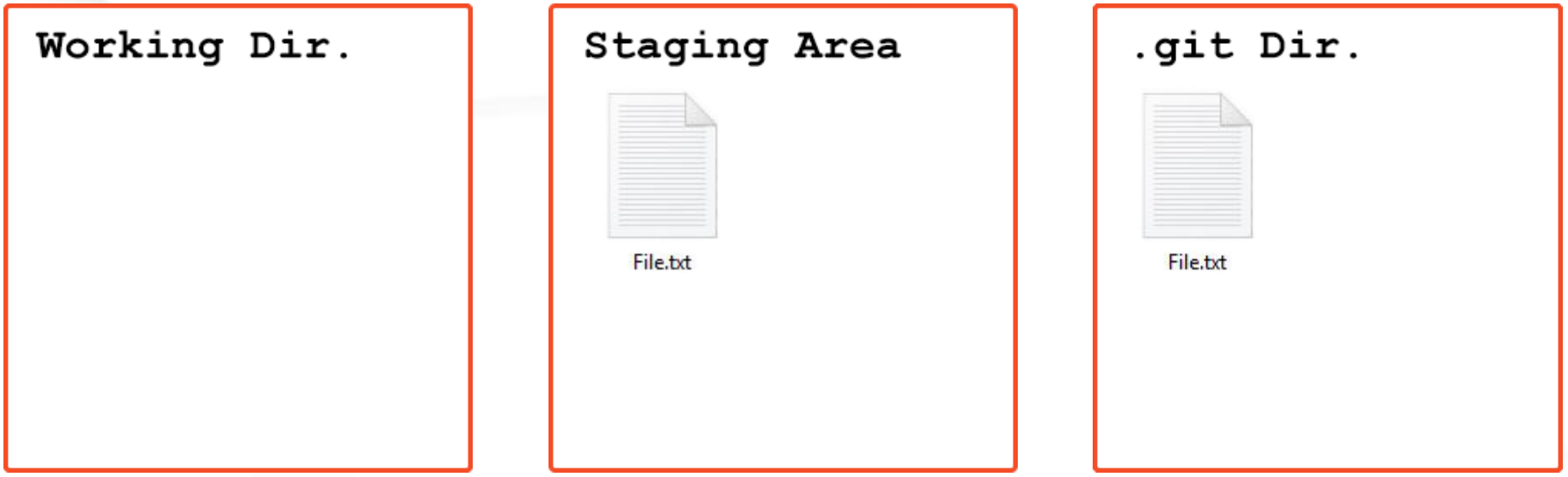
Working Dir.

Staging Area

.git Dir.

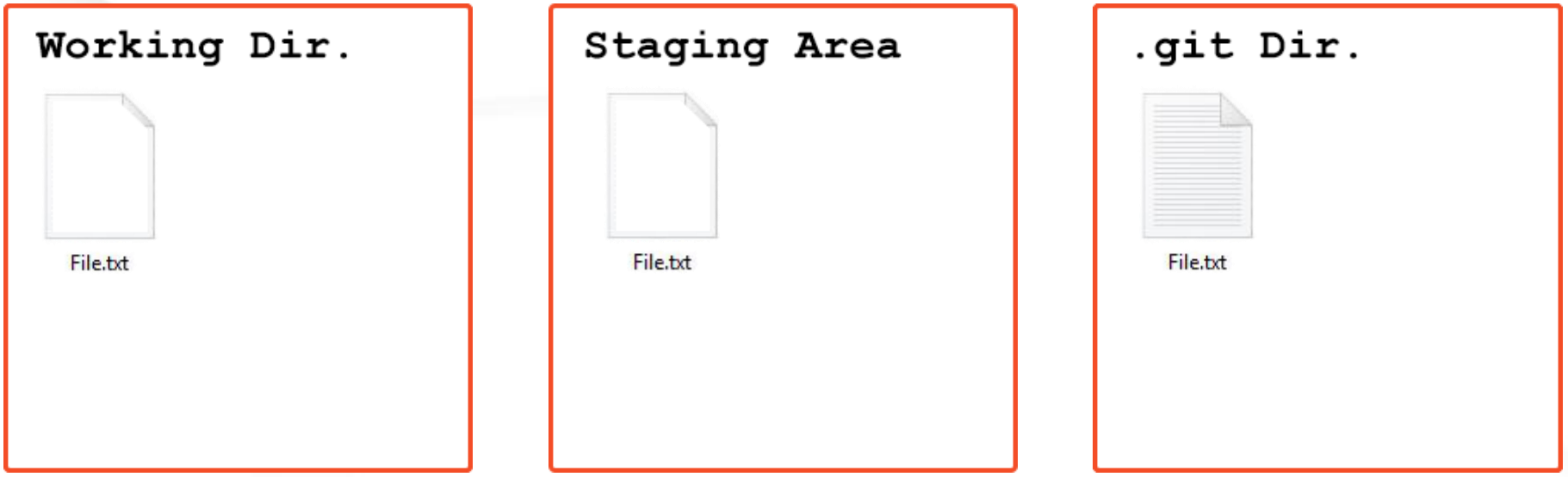
Comandos Básicos

git status



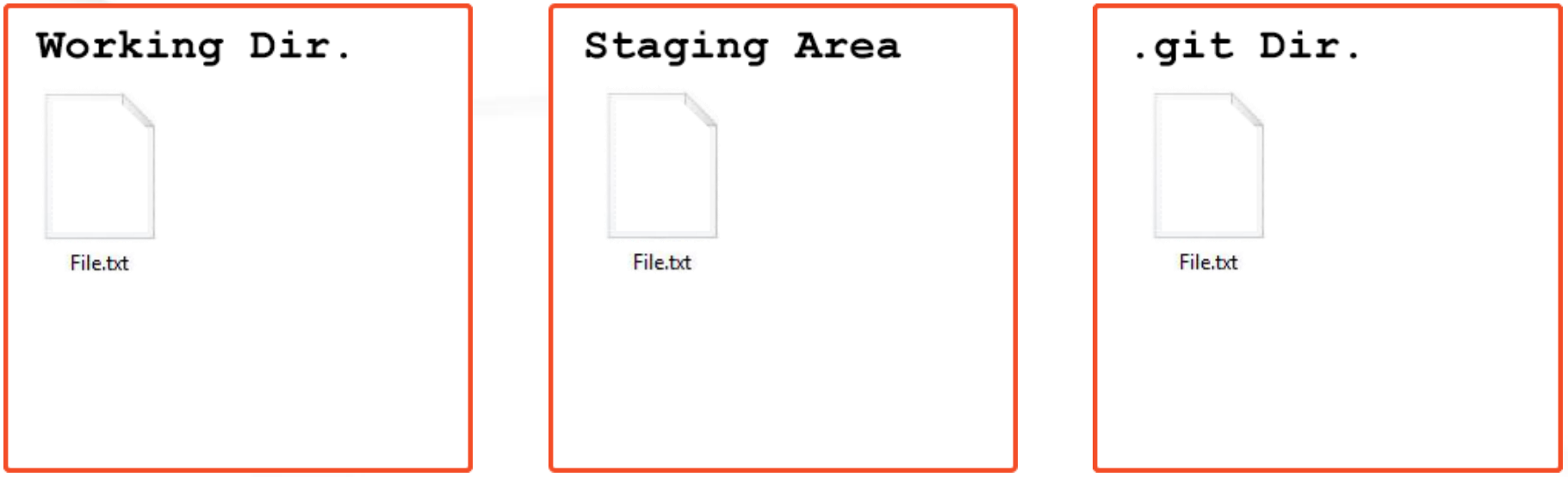
Comandos Básicos

```
git add File.txt
```



Comandos Básicos

```
git commit -m “Deletei File.txt”
```



Comandos Básicos

```
git rm File.txt
```

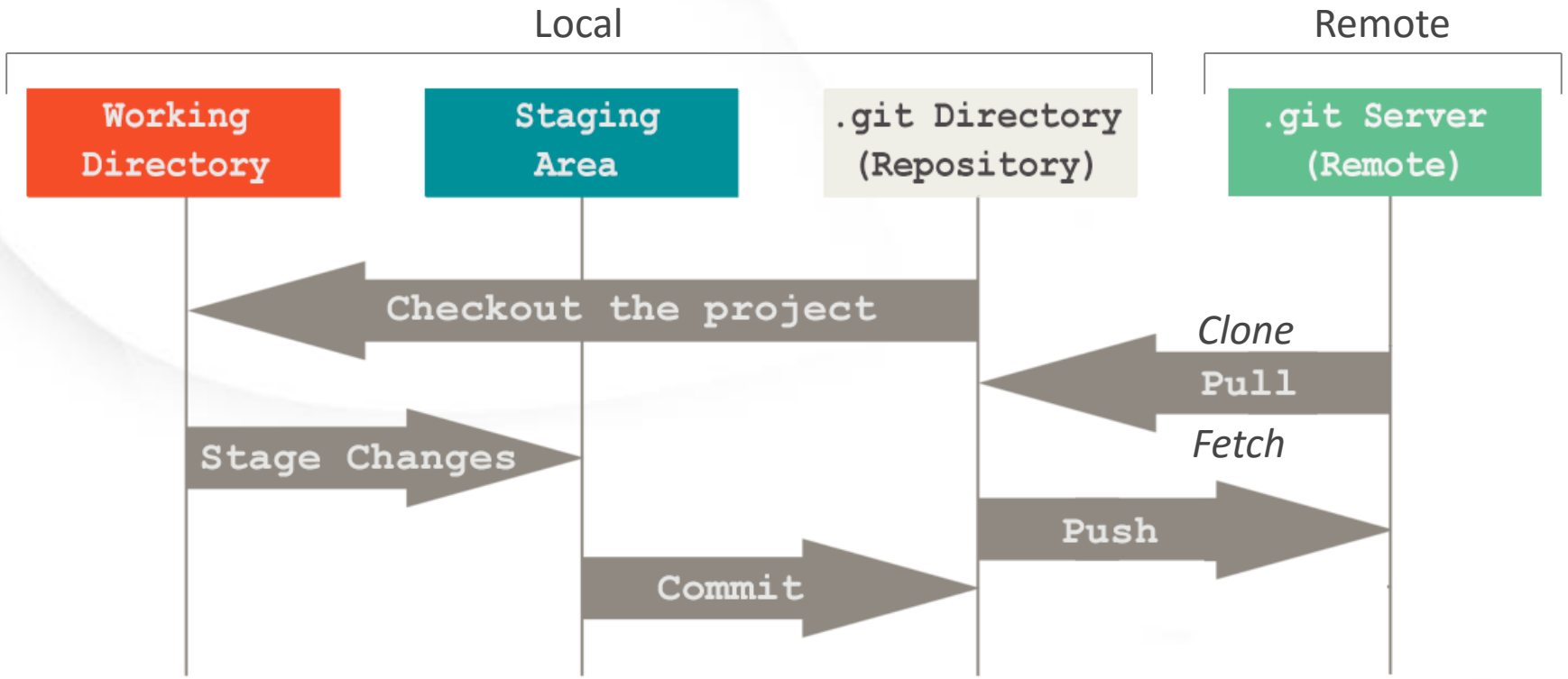
Outros Comandos

- git commit -a -m “Também adiciona as mudanças”
- git reset HEAD File.txt
- git rm --cached File.txt
- git checkout -- File.txt
- git diff

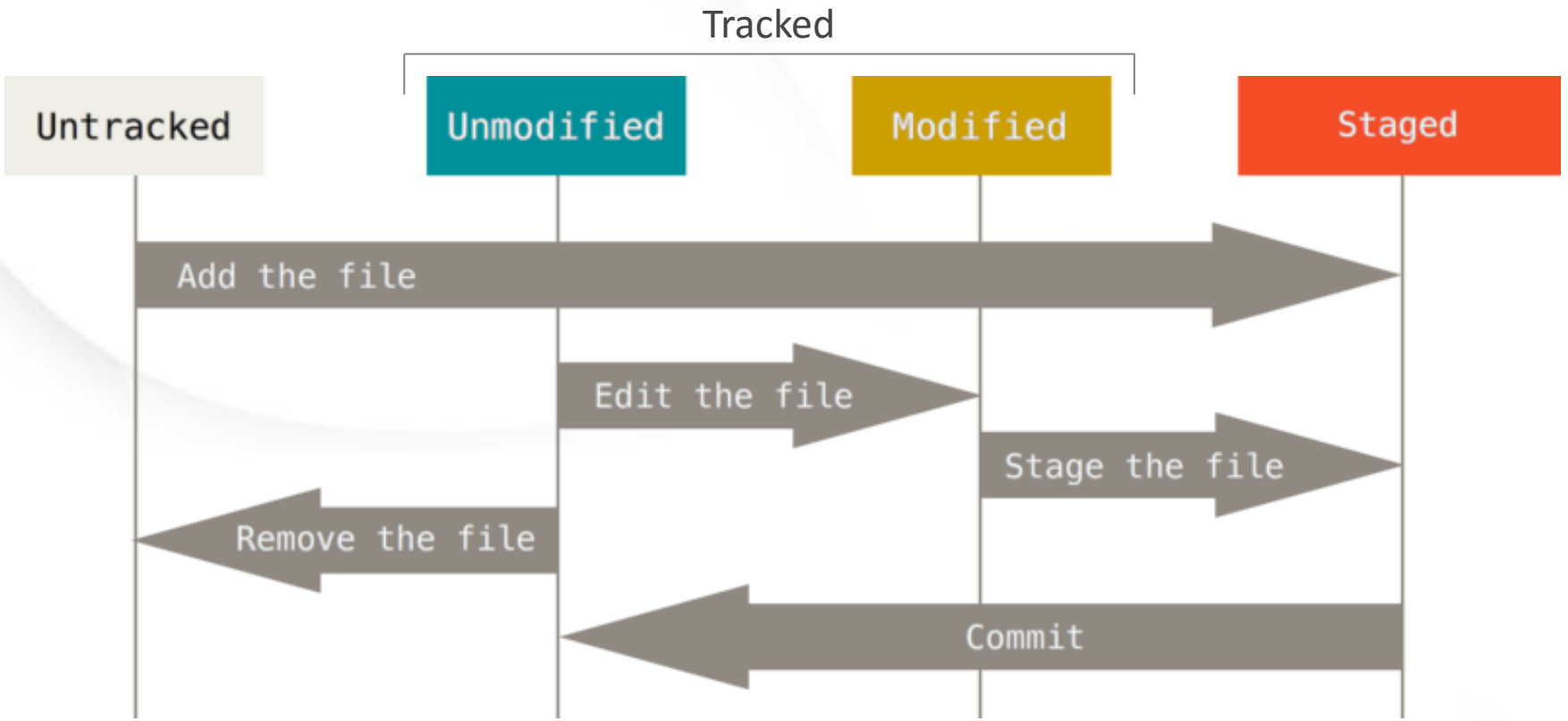
Aliases

- git config --global alias.cm commit
- git config --global alias.unstage 'reset HEAD --‘
- git config --global alias.last 'log -1 HEAD'

Workflow


Local
Remote
Working Directory
Staging Area
.git Directory (Repository)
.git Server (Remote)
Checkout the project
Clone
Pull
Fetch
Stage Changes
Push
Commit

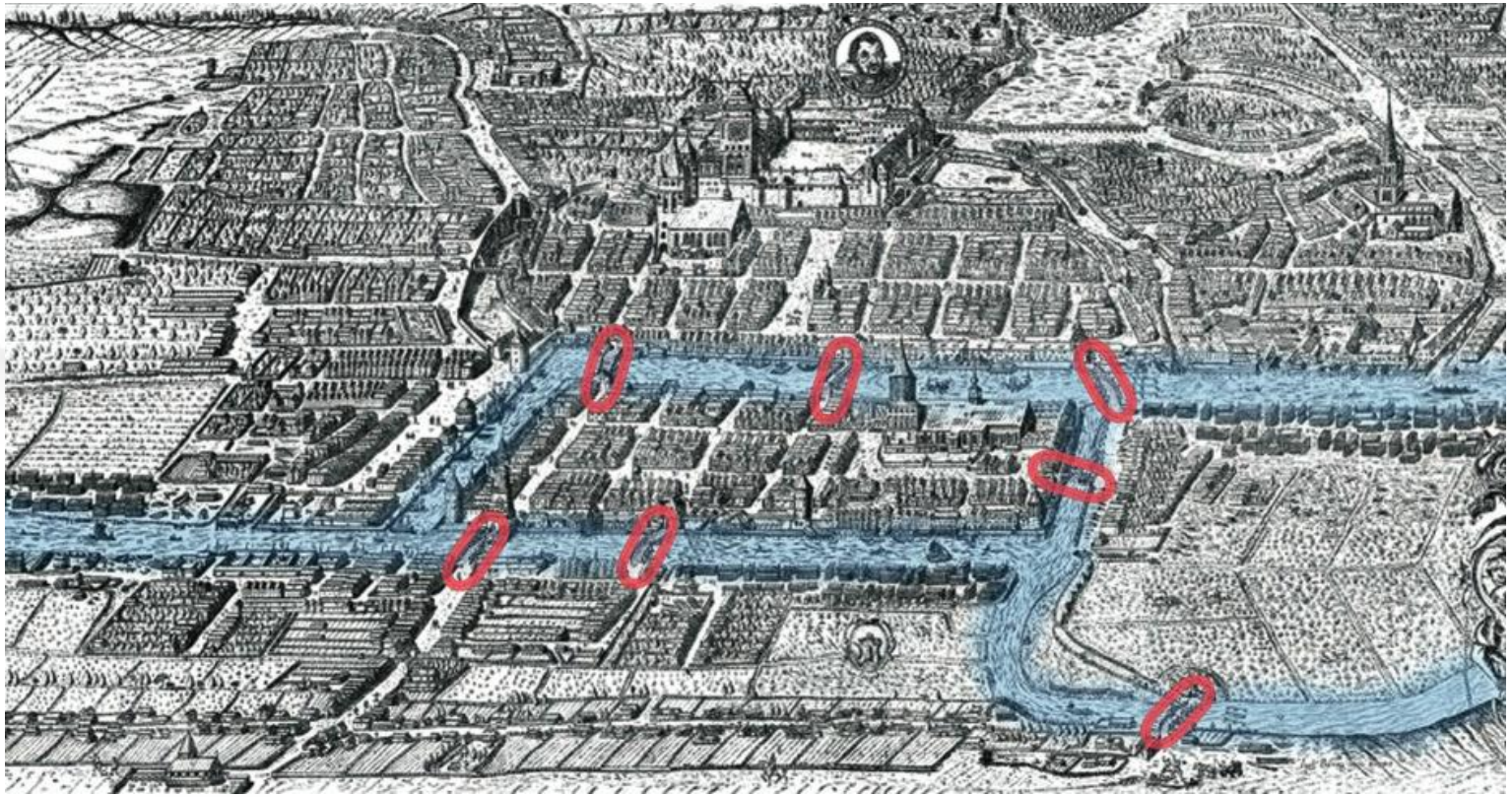
Ciclo de vida de um arquivo



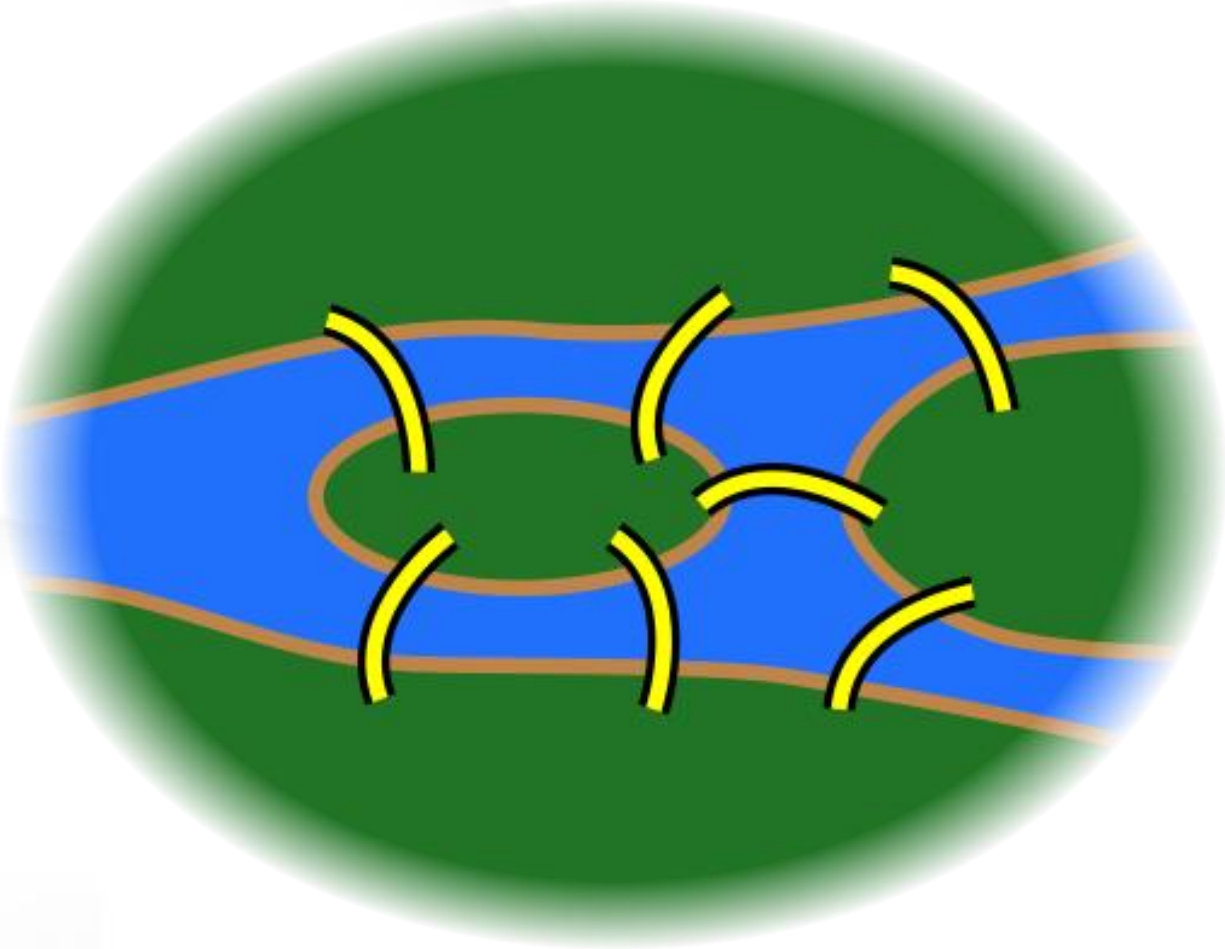
Branching

Laboratório de
Tecnologia da Informação
Aplicada

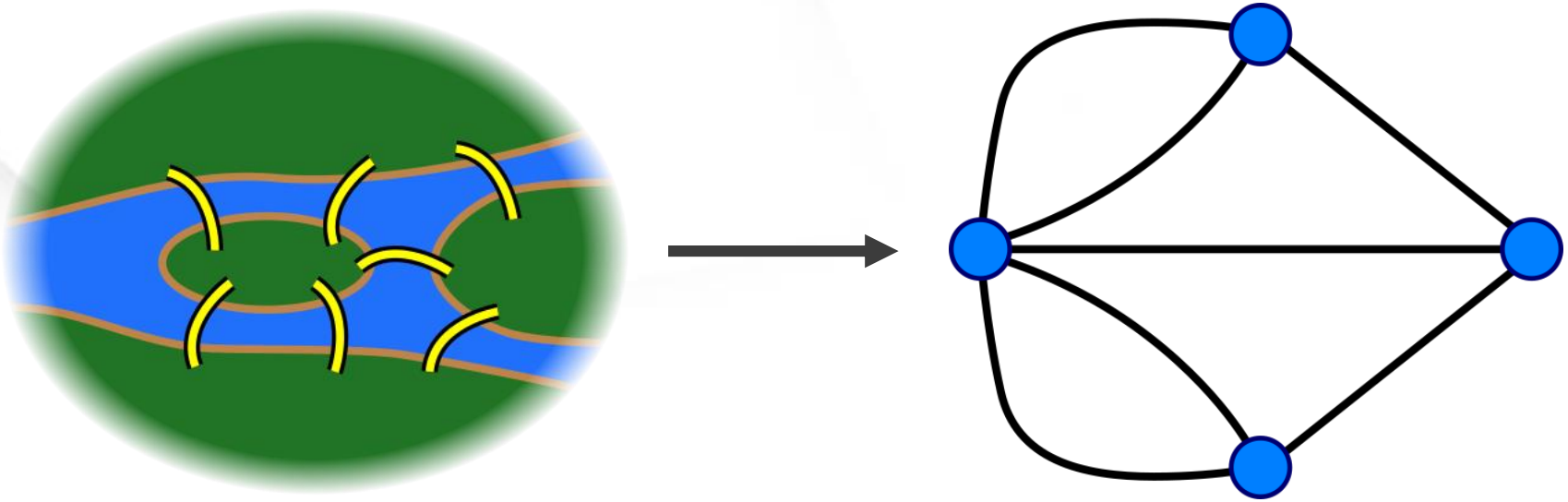
Grafos



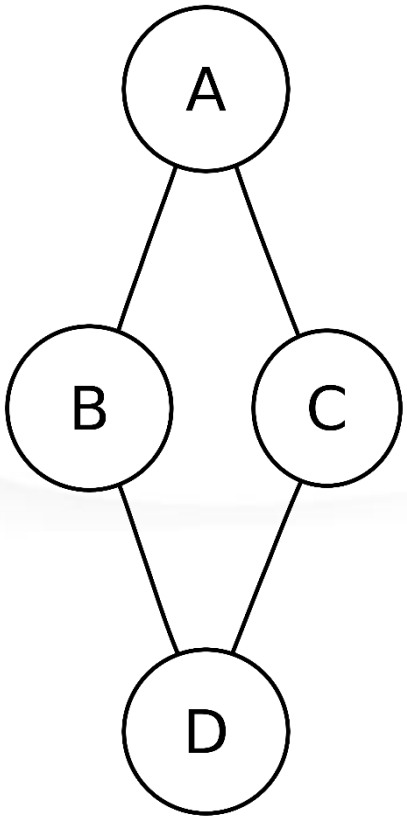
Grafos



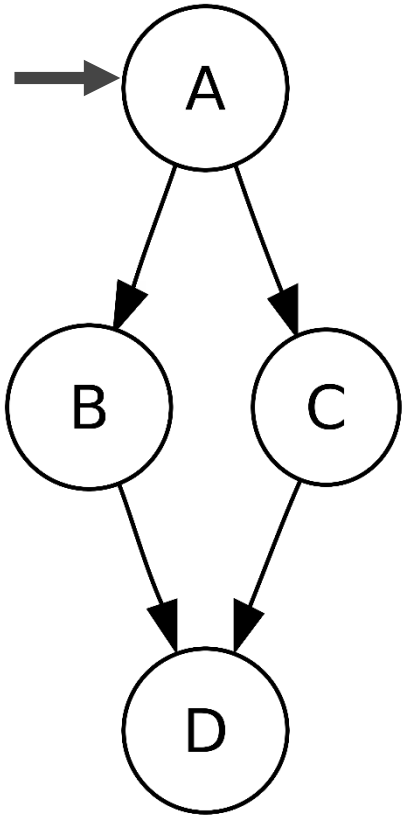
Grafos



Grafos

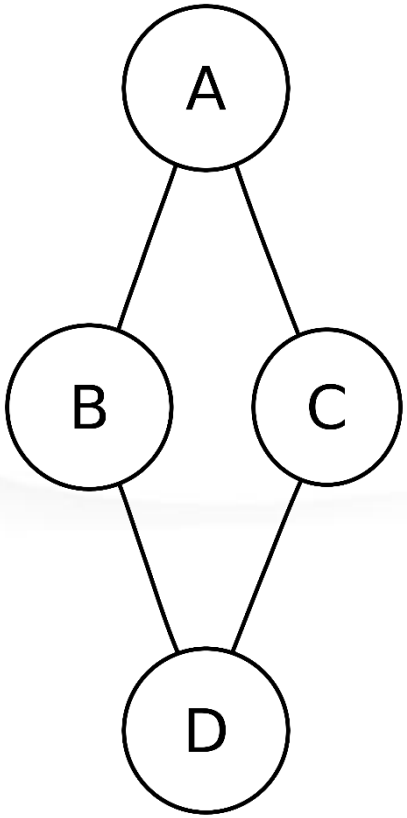


Não Direcionado

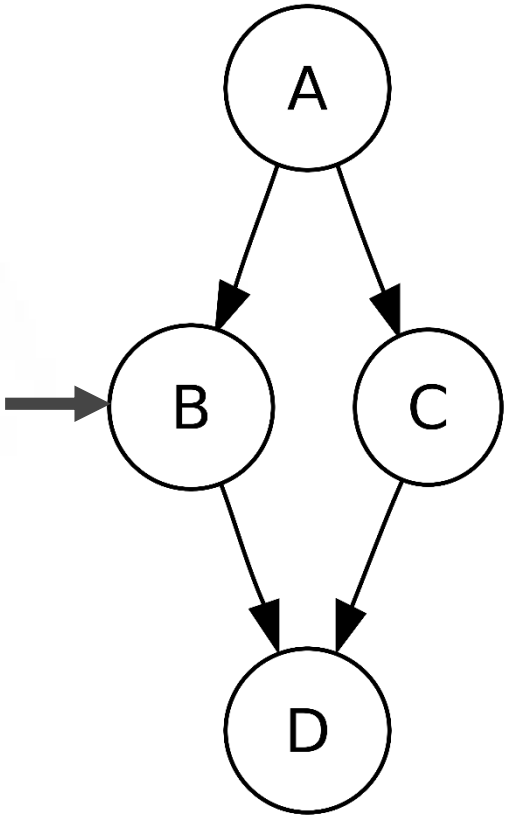


Direcionado

Grafos

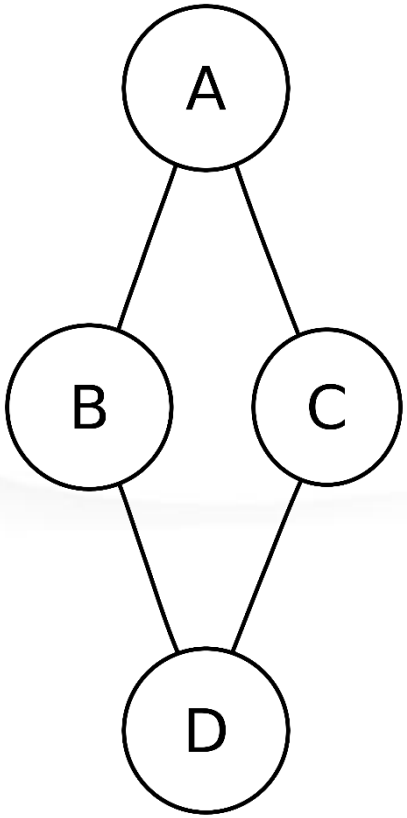


Não Direcionado

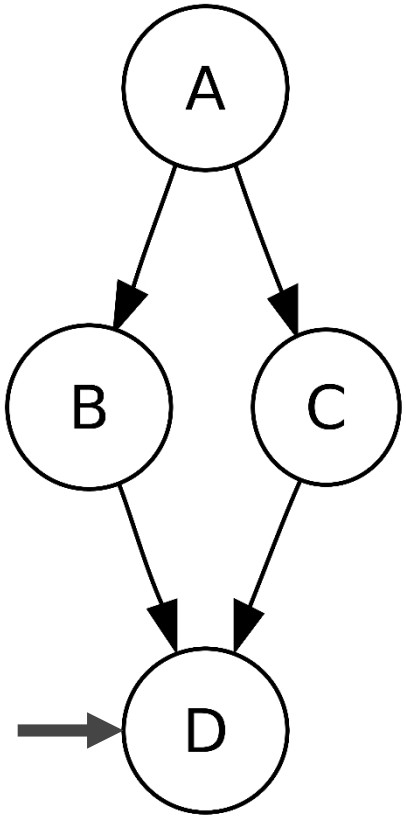


Direcionado

Grafos

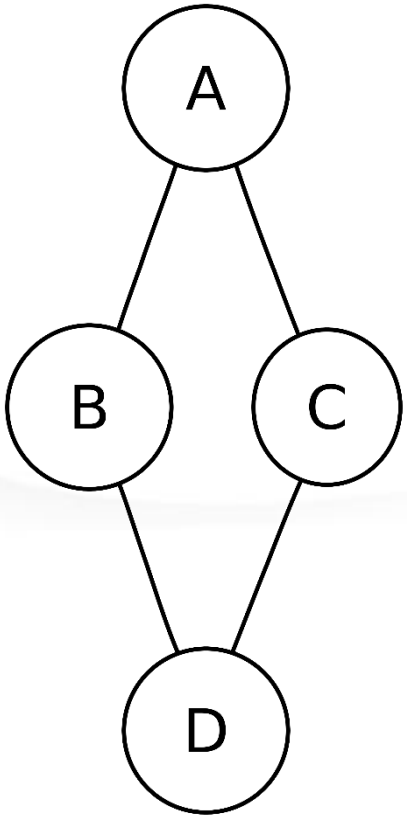


Não Direcionado

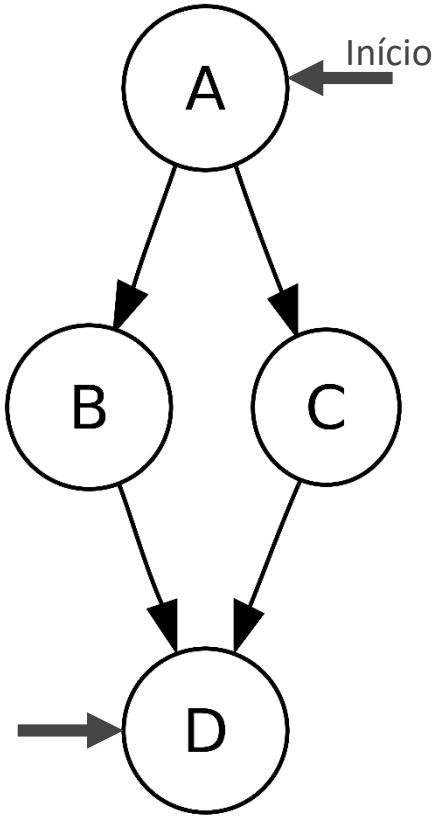


Direcionado

Grafos



Não Direcionado



Direcionado

Branching

Criar um branch significa dizer que você vai divergir da linha principal de desenvolvimento e continuar a trabalhar sem bagunçar essa linha principal.

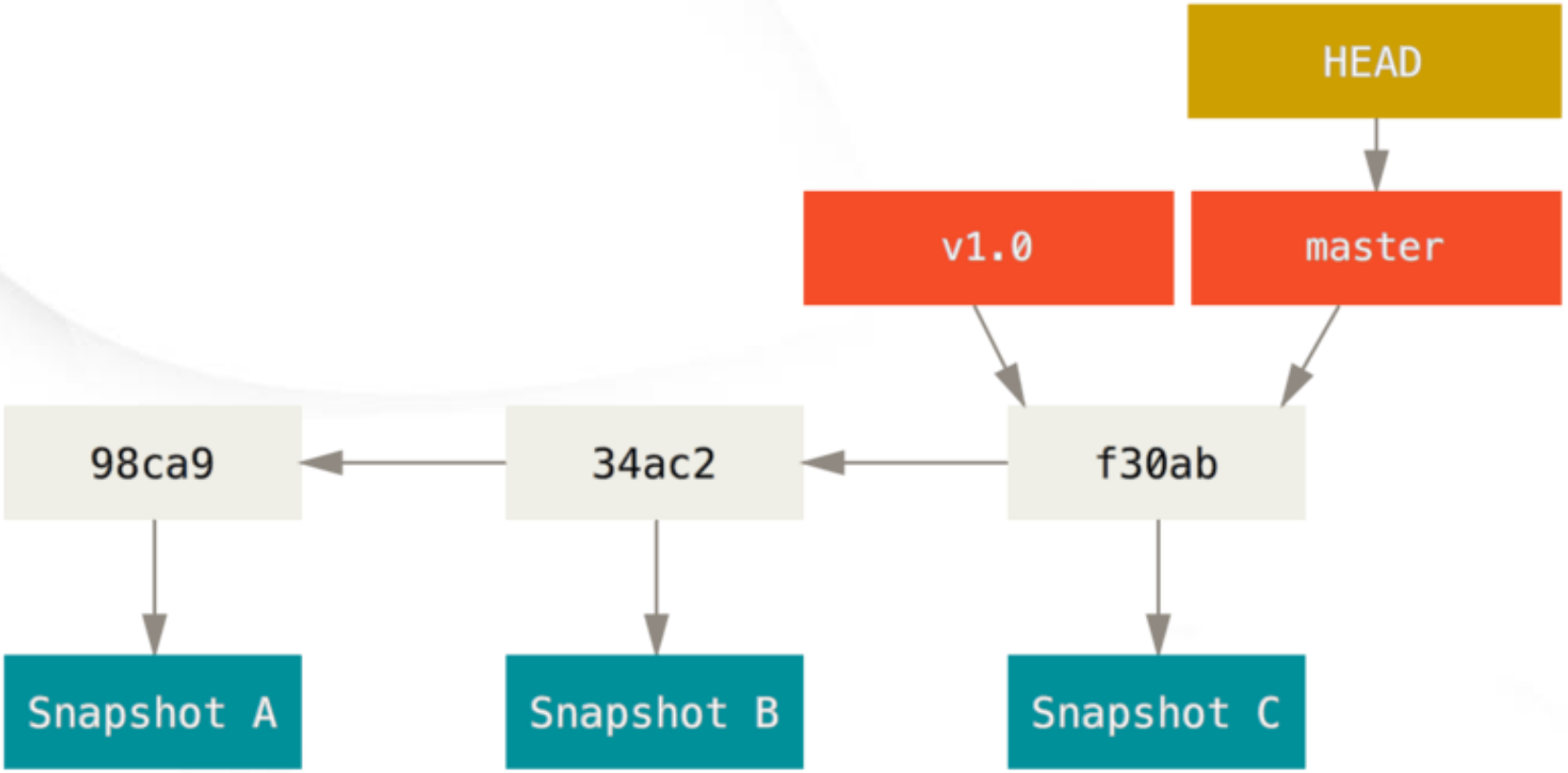
A forma como o Git cria branches é inacreditavelmente leve, fazendo com que as operações sejam praticamente instantâneas e a alternância entre os branches seja tão rápida quanto.

O Git incentiva um fluxo de trabalho no qual se fazem branches e merges com frequência, até mesmo várias vezes ao dia.

Pro Git, Scott Chacon e Ben Straub

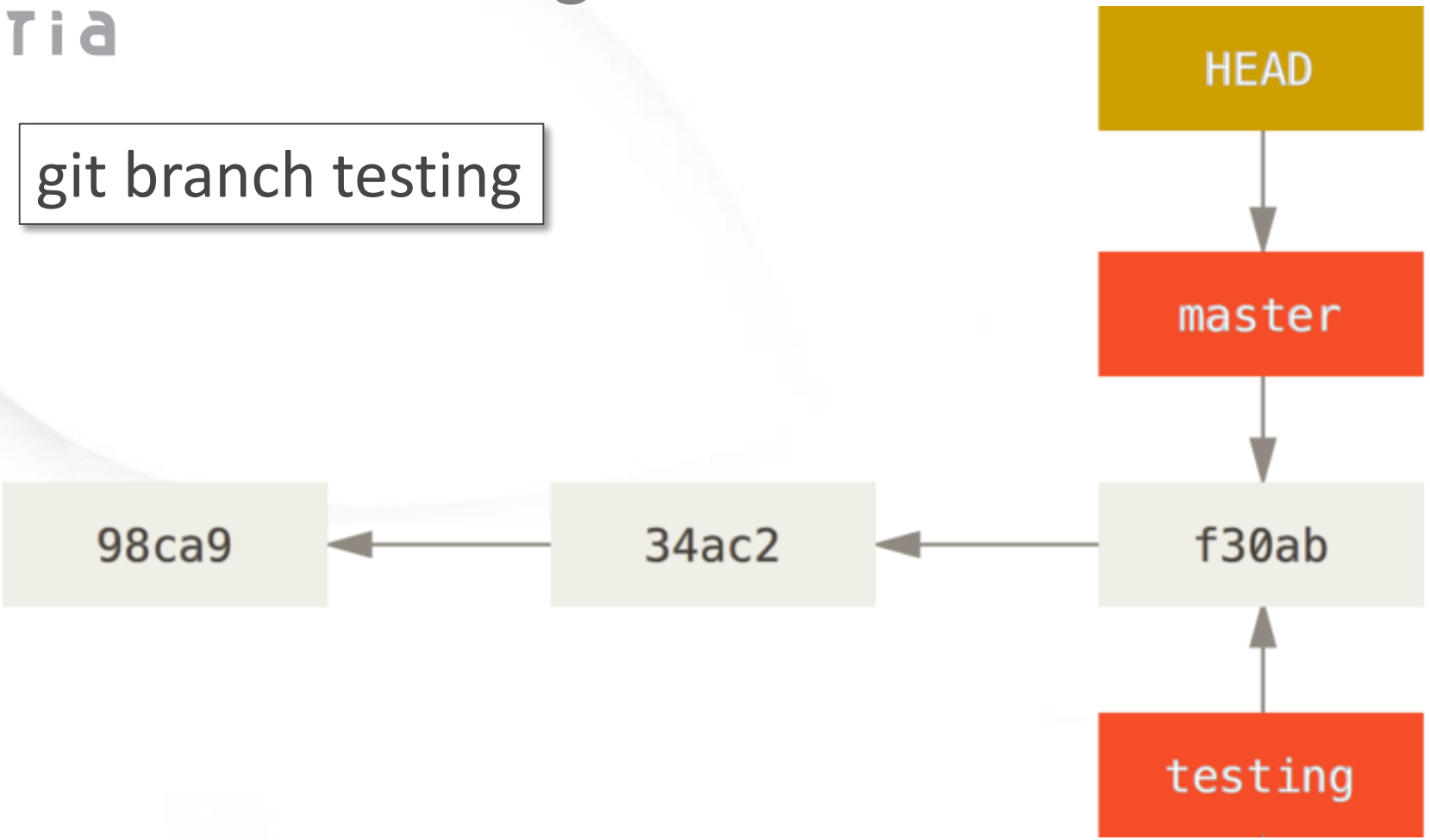

LTia

Branching


HEAD
v1.0
master
98ca9
34ac2
f30ab
Snapshot A
Snapshot B
Snapshot C

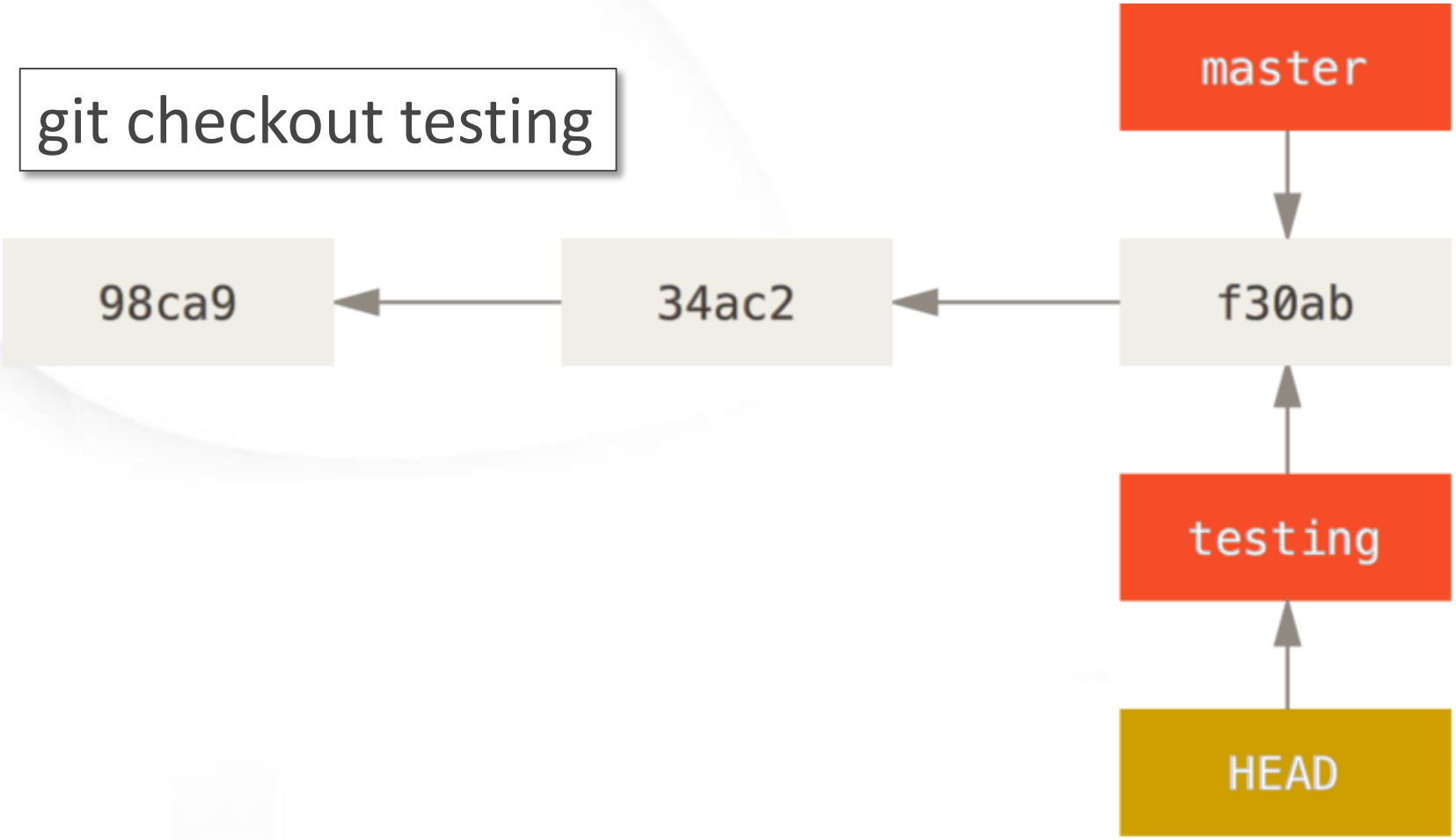
Branching

git branch testing



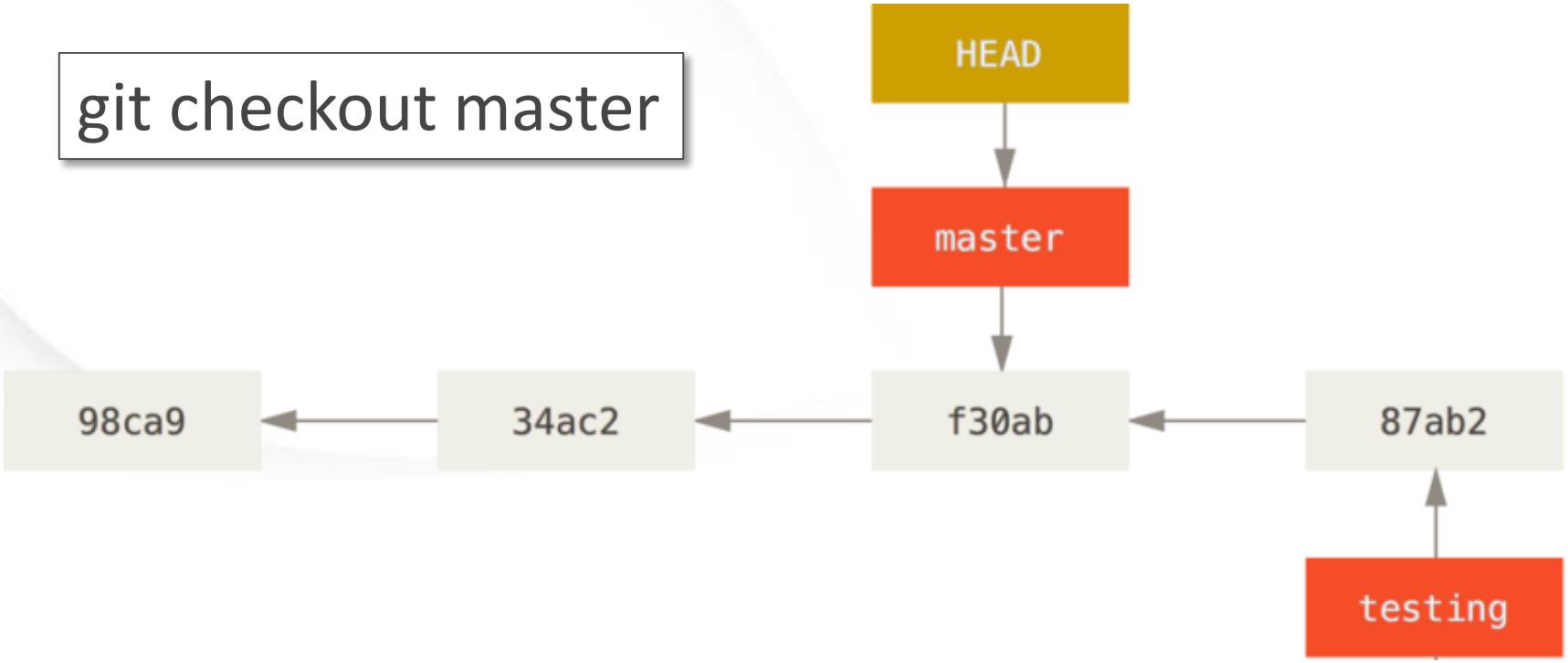
Branching

git checkout testing

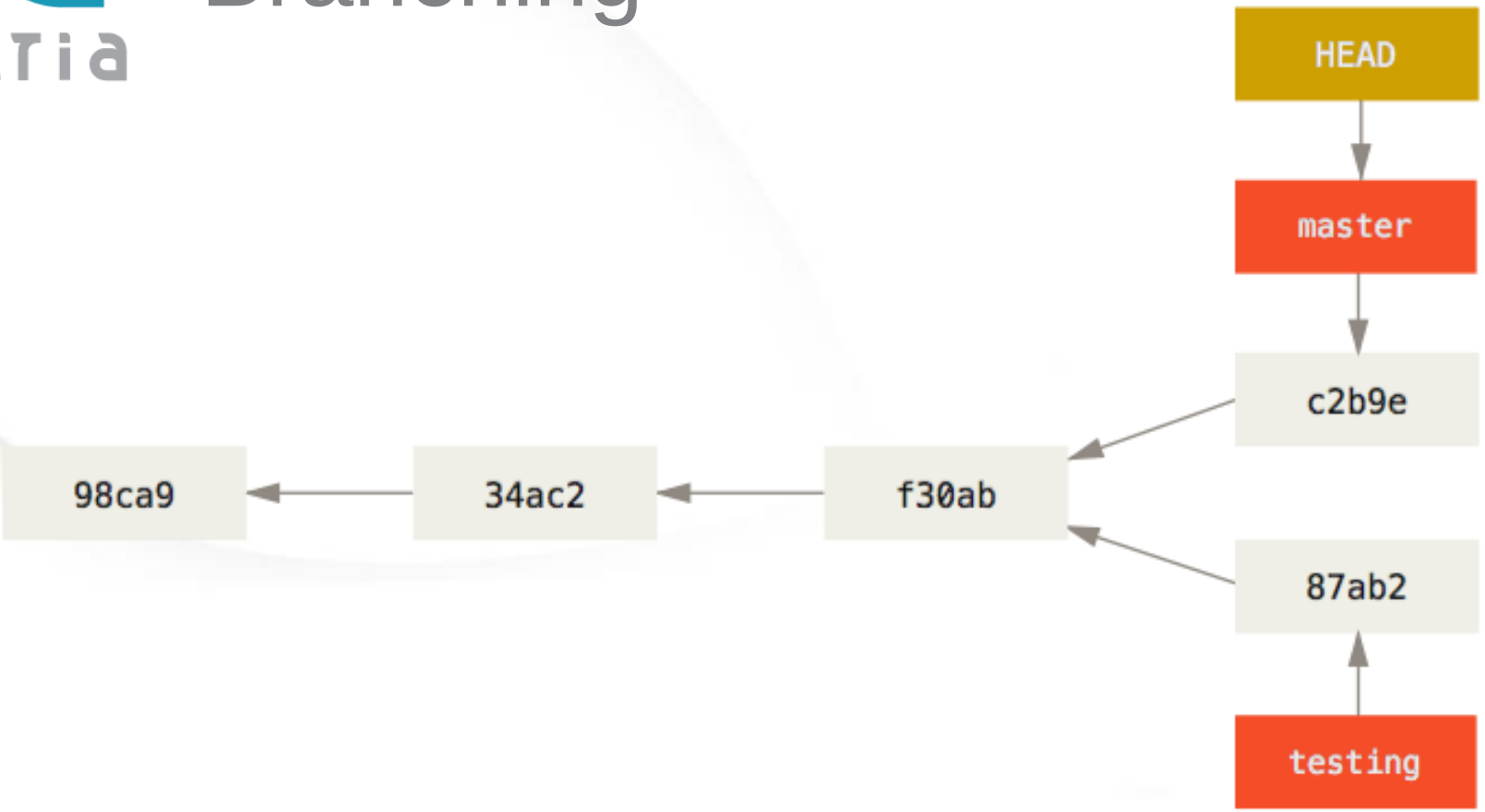


Branching

git checkout master

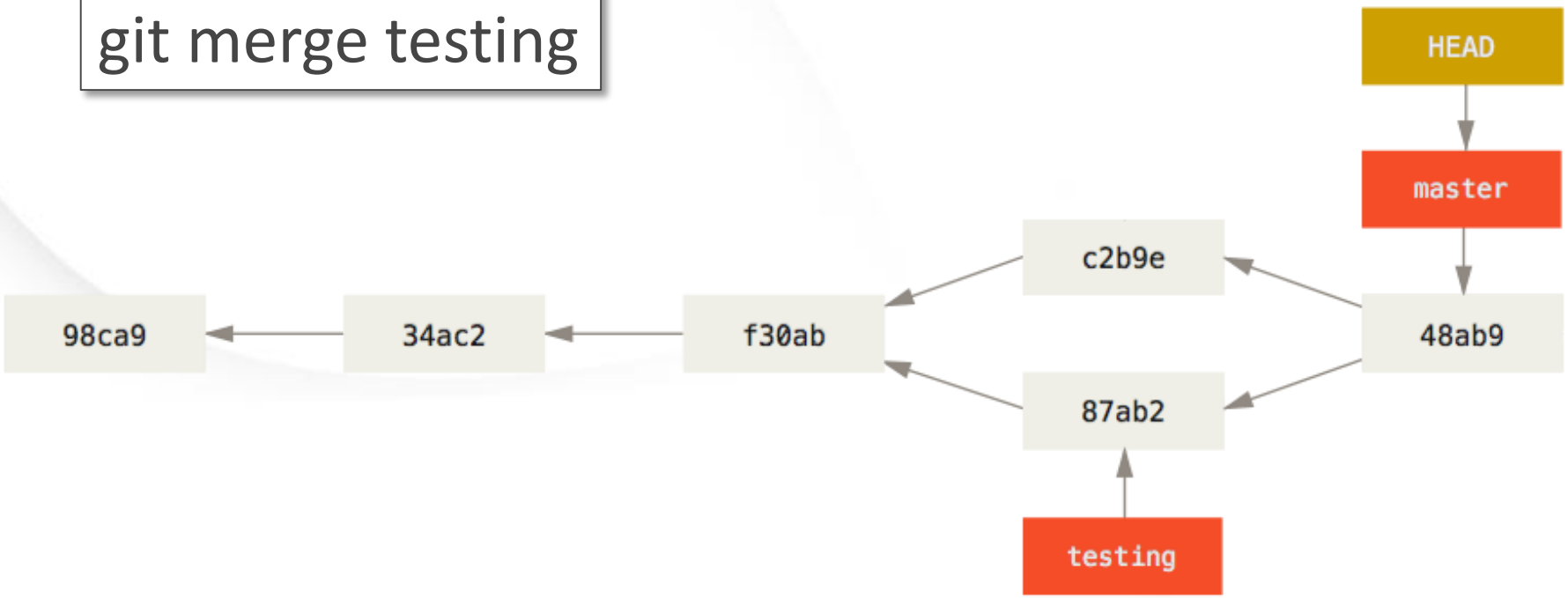


Branching



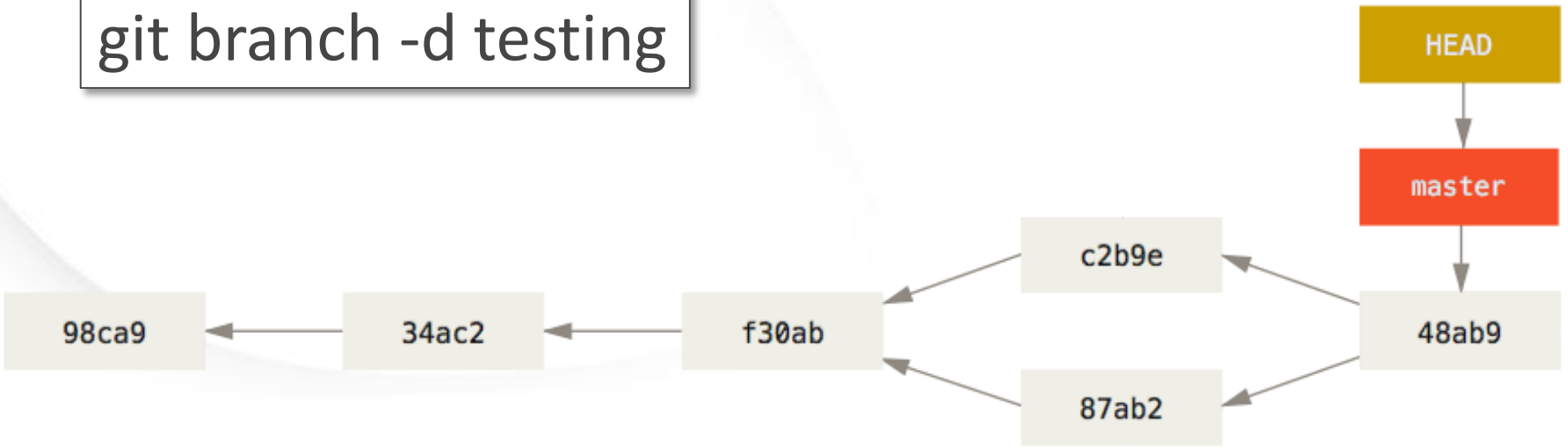
Branching

git merge testing



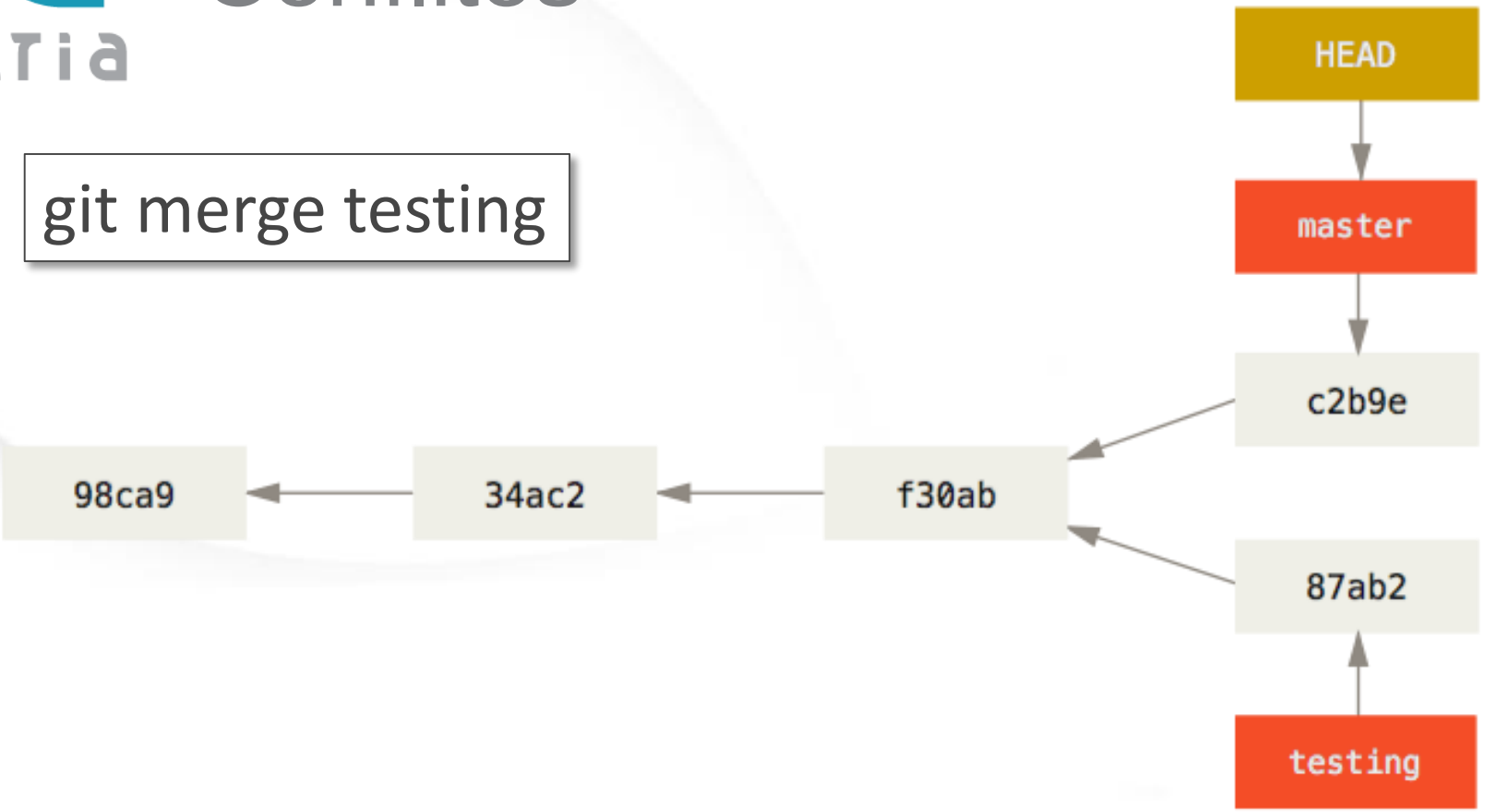
Branching

```
git branch -d testing
```



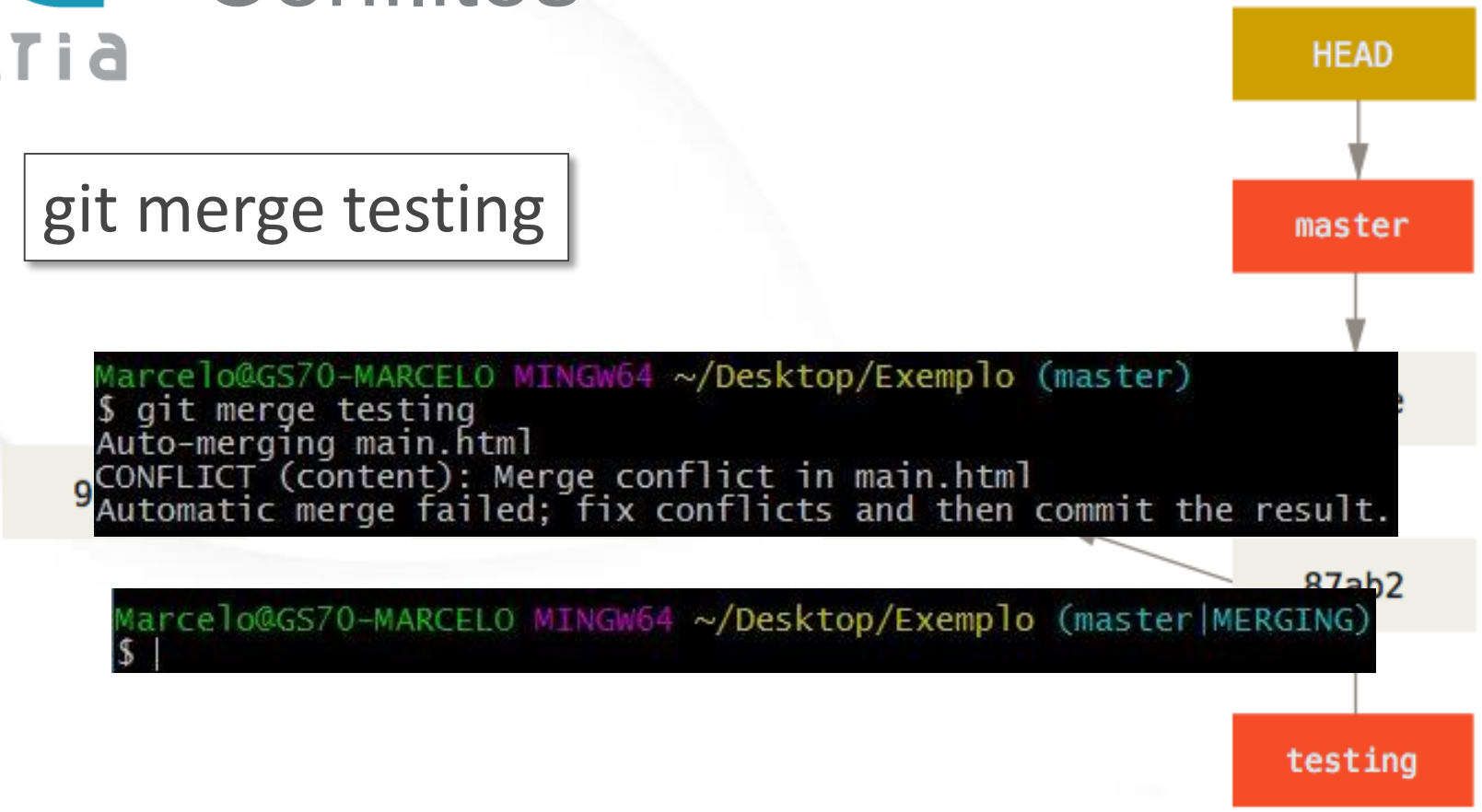
Conflitos

git merge testing



Conflitos

git merge testing



Conflitos

```html
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="style.css" />
  </head>
  <body>
  </body>
</html>
```

```html
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="estilo.css" />
  </head>
  <body>
  </body>
</html>
```

Conflitos

```
1  <html>
2    <head>
3  <<<<<<< HEAD
4      <link type="text/css" rel="stylesheet" media="all" href="estilo.css" />
5  =======
6      <link type="text/css" rel="stylesheet" media="all" href="style.css" />
7  >>>>>>> testing
8    </head>
9    <body>
10   </body>
11 </html>
```

Conflitos

```
1 <html>
2   <head>
3     <link type="text/css" rel="stylesheet" media="all" href="style.css" />
4   </head>
5   <body>
6   </body>
7 </html>
```

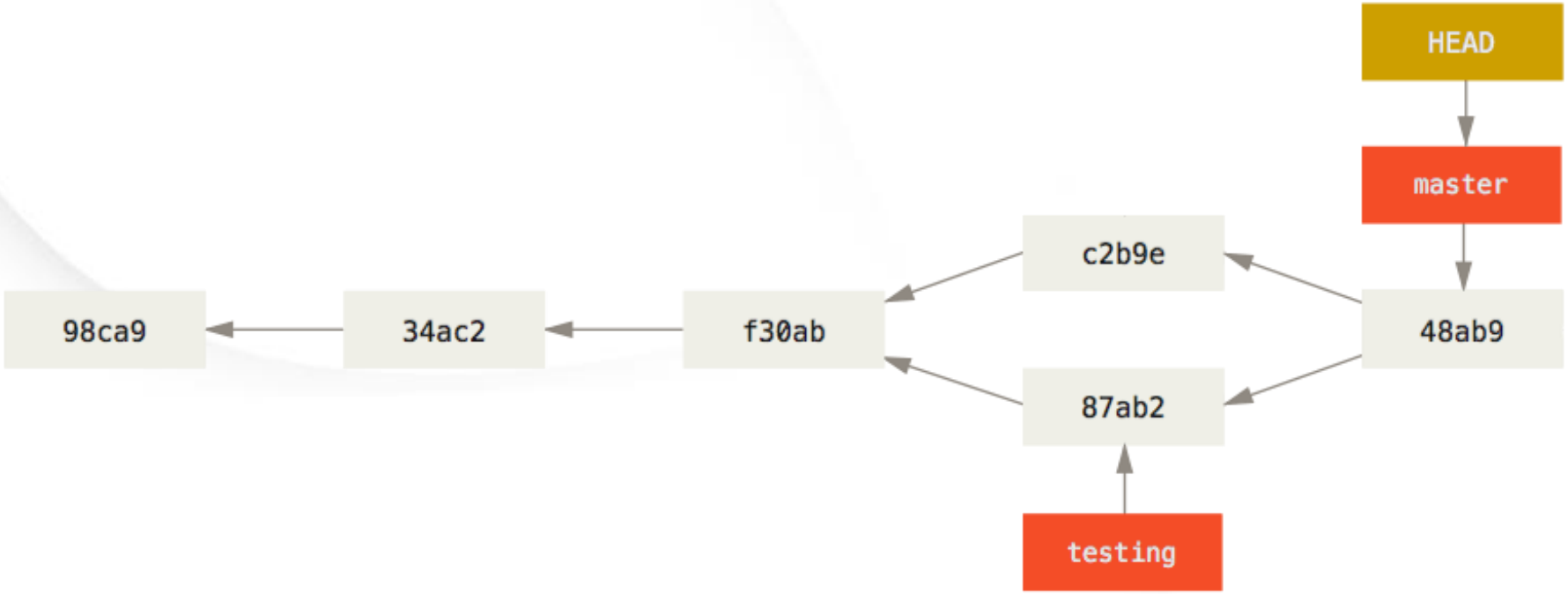
Conflitos

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master|MERGING)
$ git add main.html
```

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master|MERGING)
$ git commit -m "Merged com testing, conflito resolvido"
[master 0df50af] Merged com testing, conflito resolvido
```

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master)
$
```

Conflitos

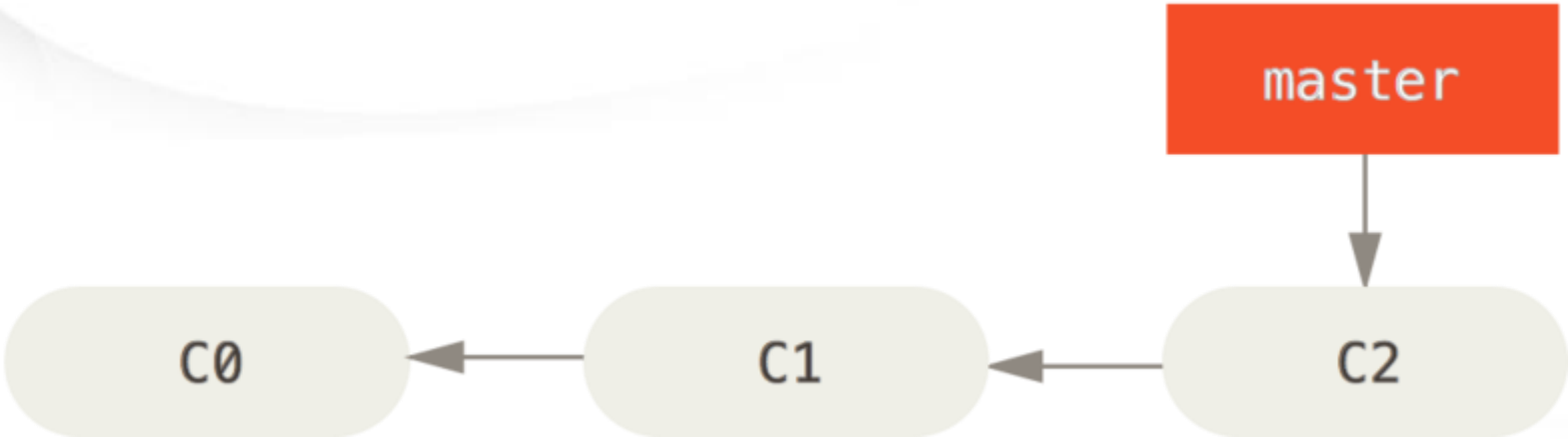


Tags

- git tag -a v1.0 -m “Versão 1.0”
- git tag -a v2.0 -m 98ca9 “Versão 2.0”

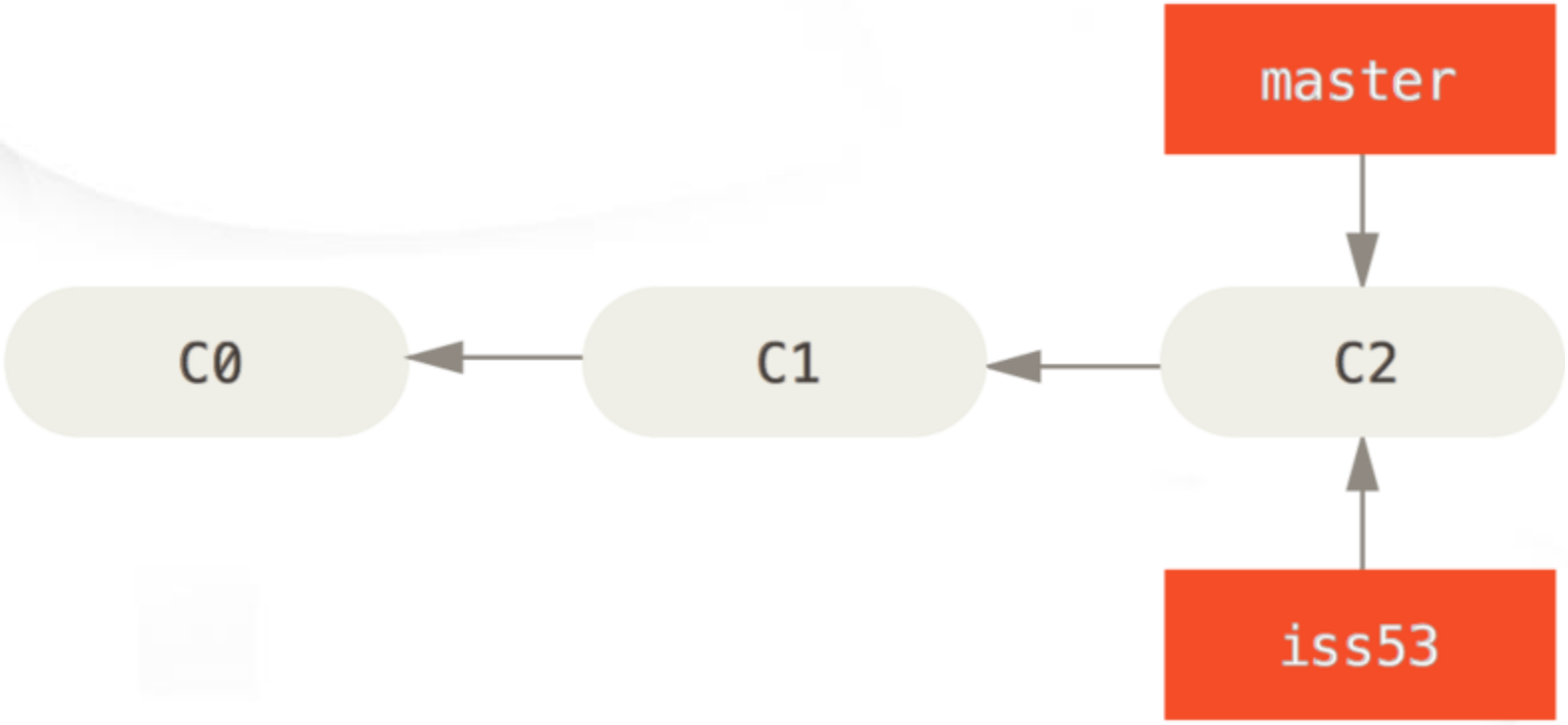
Branching - Exercício

Suponha que você está trabalhando em um site. Inicialize e configure o diretório, crie um arquivo e faça três commits com ele, alterando algo no arquivo entre cada commit.



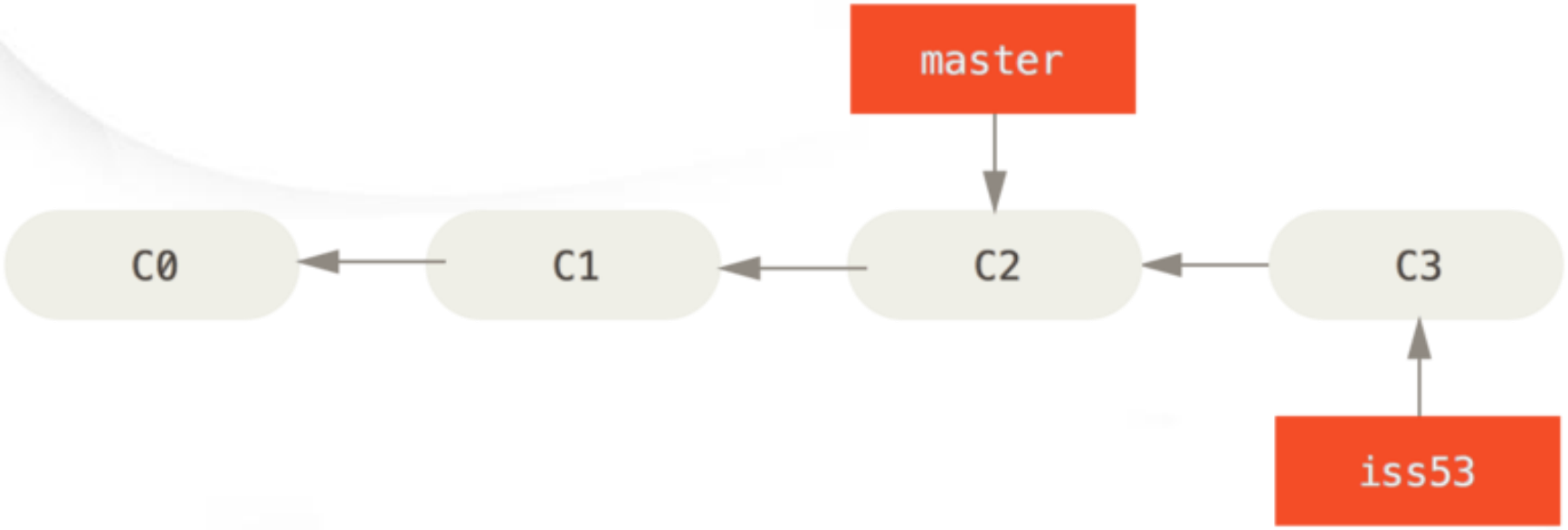
Branching - Exercício

Agora você decide resolver o Issue #53, portanto crie um branch para ele (*iss53*).



Branching - Exercício

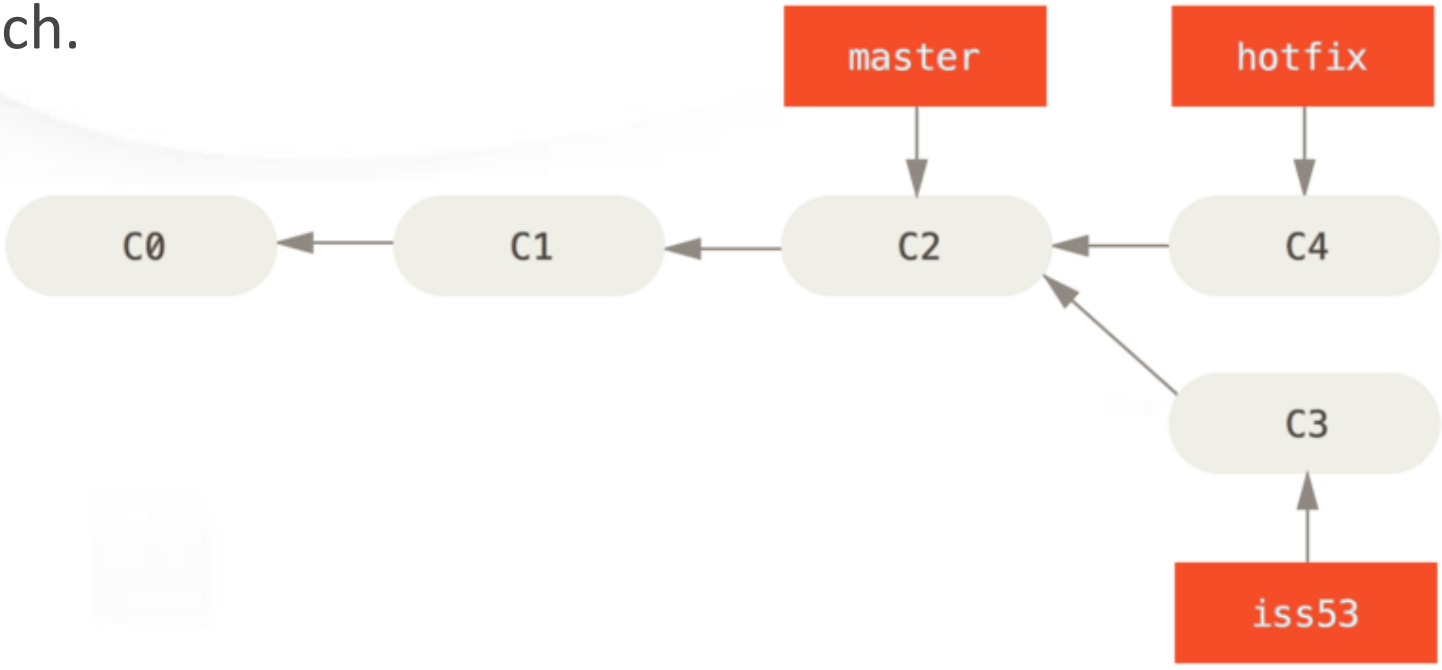
Crie outro arquivo, e adicione-o em um commit para o branch *iss53*.



Branching - Exercício

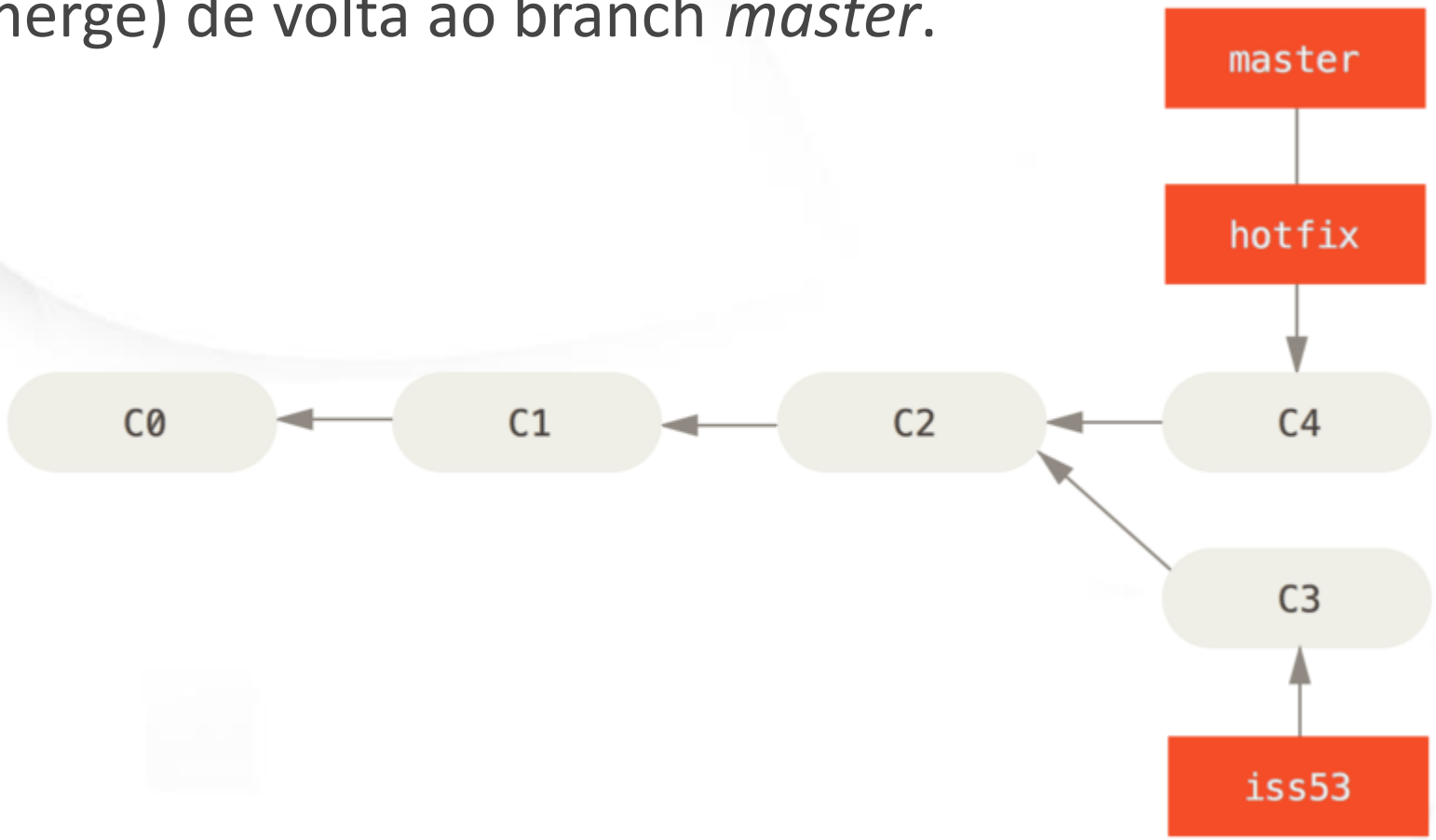
Encontraram um problema urgente no site que precisa ser resolvido antes de qualquer outra coisa.

Crie um novo branch para resolver o problema (*hotfix*), crie um novo arquivo e adicione-o em um commit para esse novo branch.



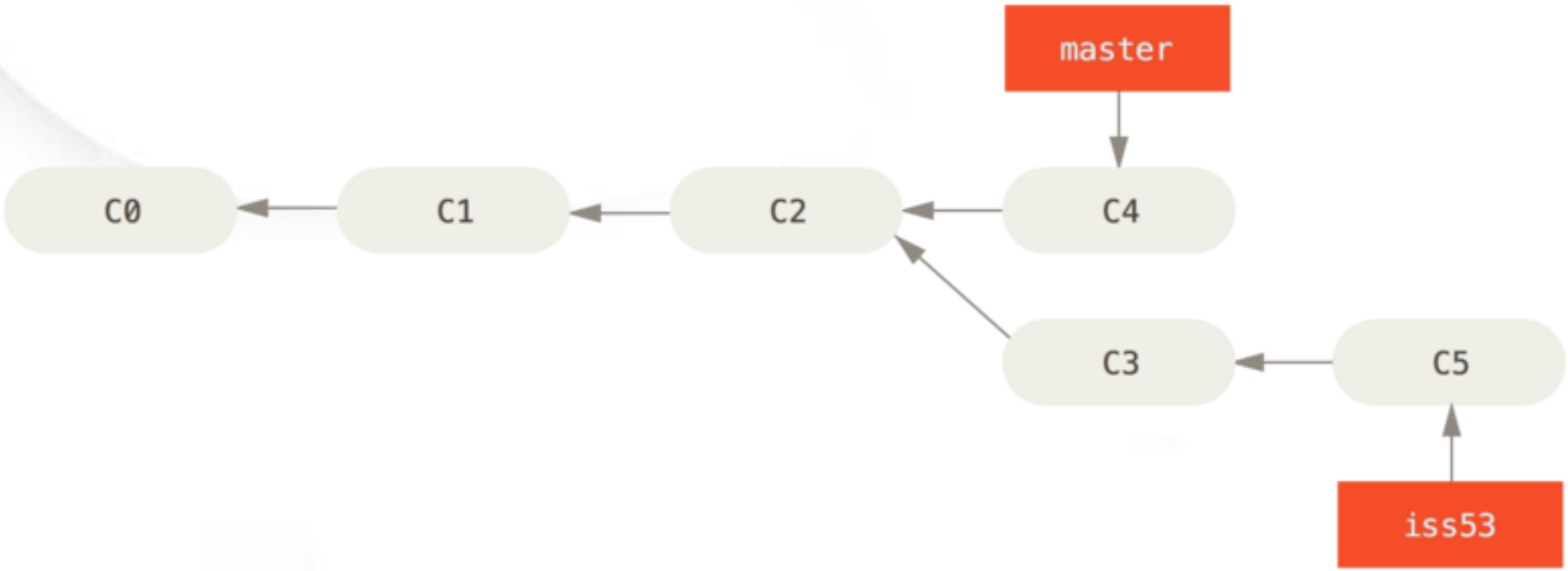
Branching - Exercício

O problema urgente foi resolvido! Incorpore ele (merge) de volta ao branch *master*.



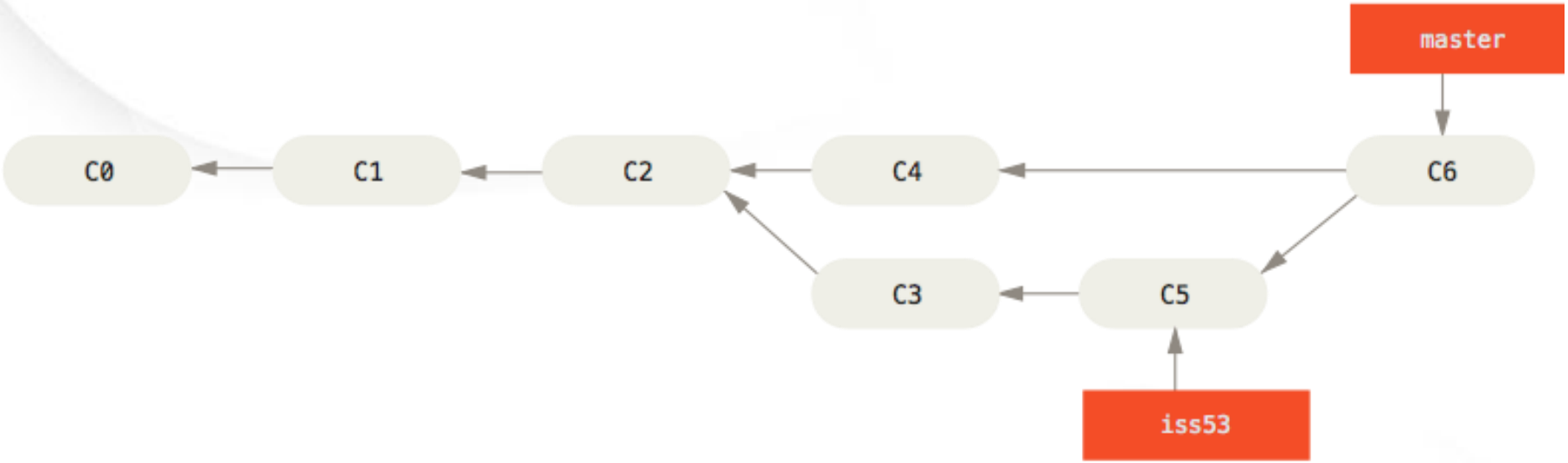
Branching - Exercício

Exclua o branch *hotfix* e volte a trabalhar no Issue #53. Altere algum arquivo e faça um commit.



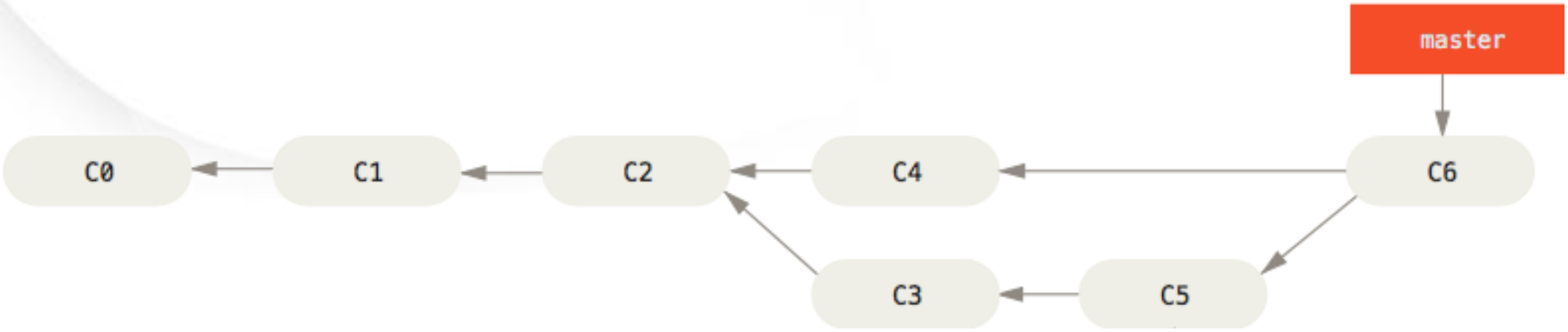
Branching - Exercício

O Issue #53 foi resolvido! Incorpore as mudanças de volta ao branch *master*.

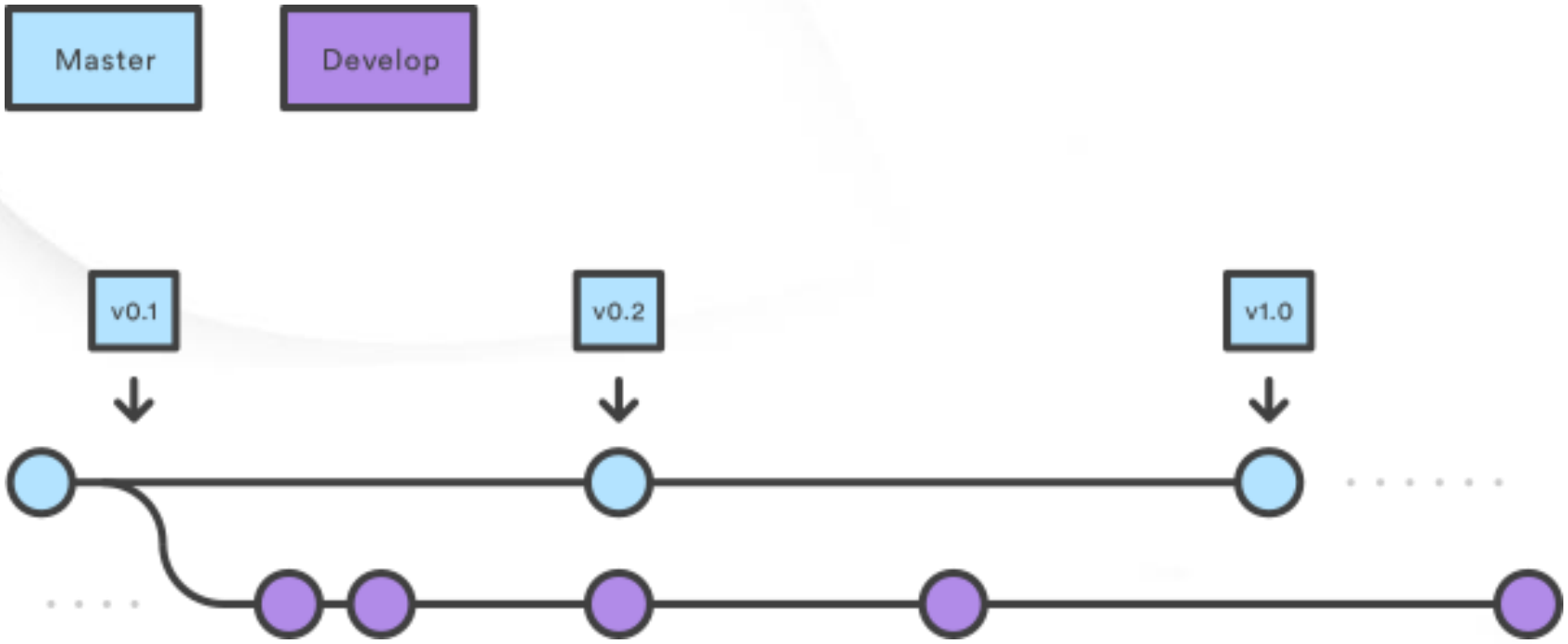


Branching - Exercício

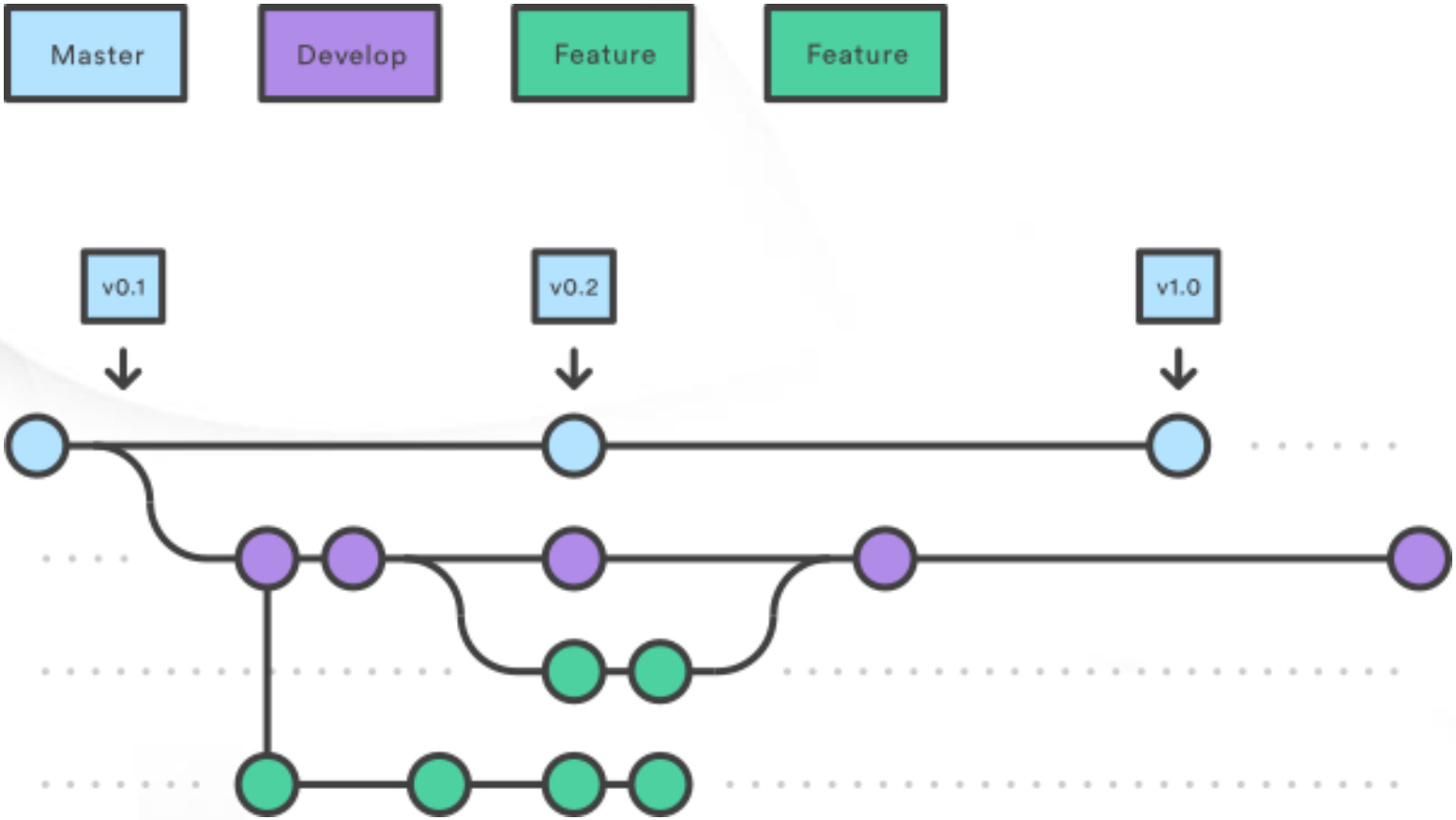
Exclua o branch *iss53*.



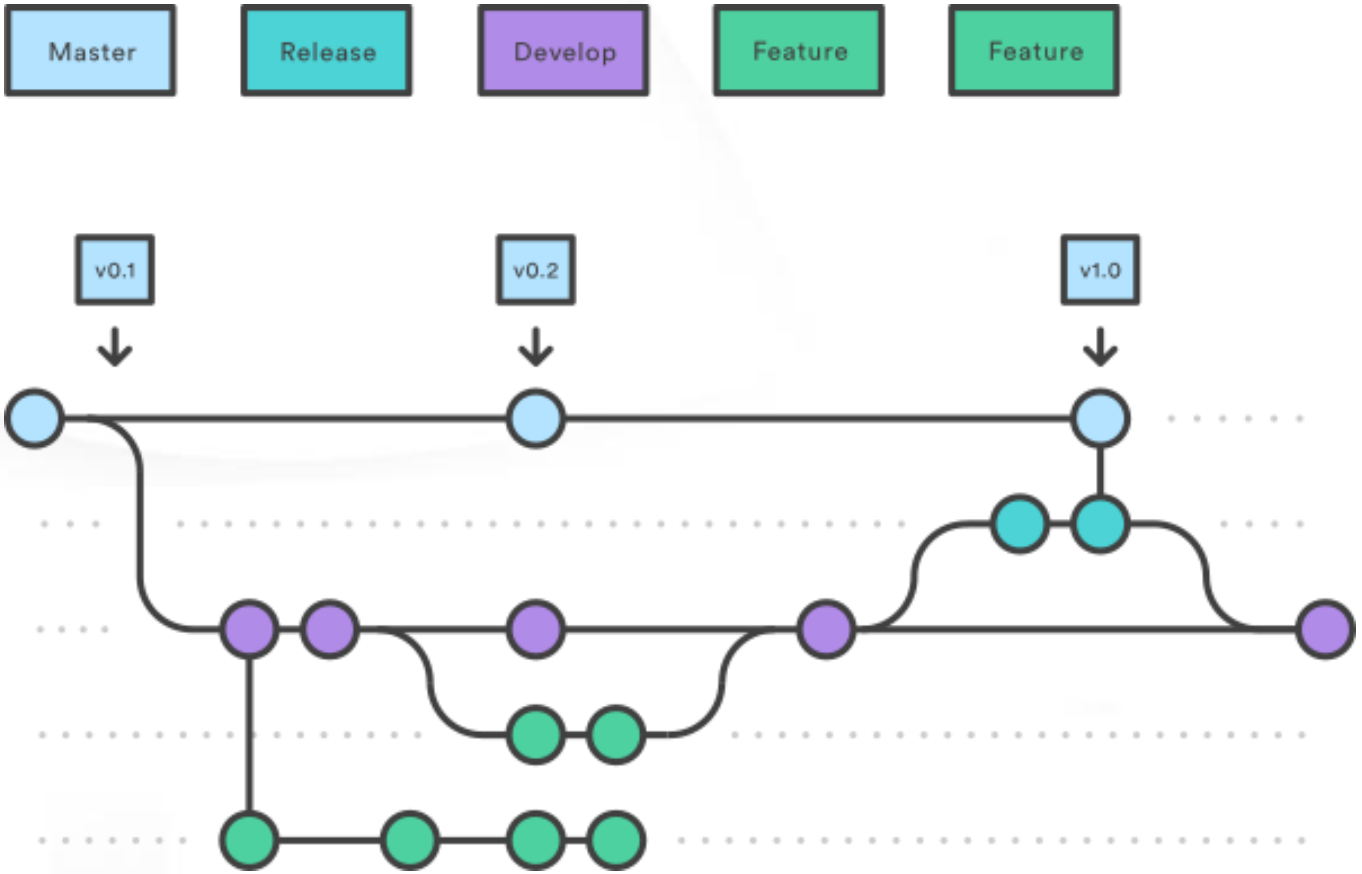
Branch Workflow



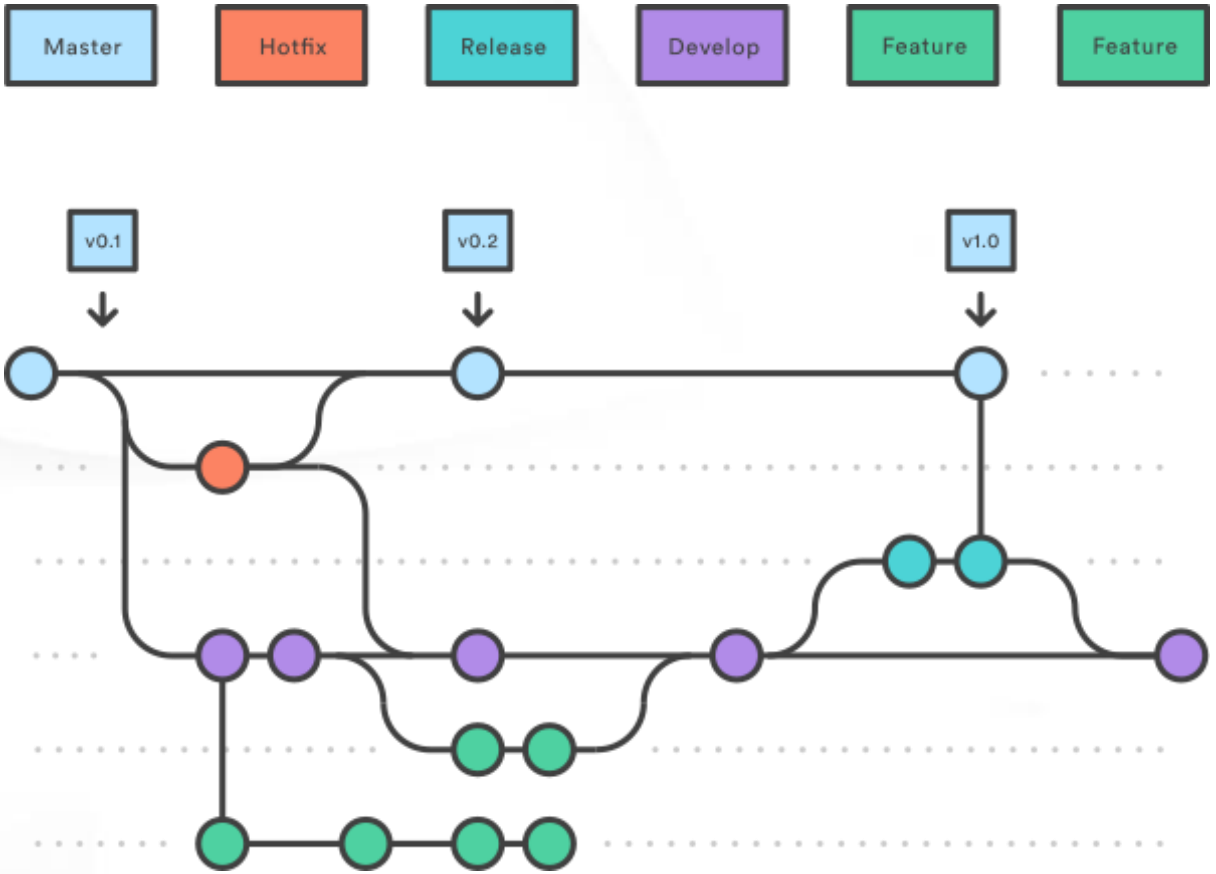
Branch Workflow



Branch Workflow



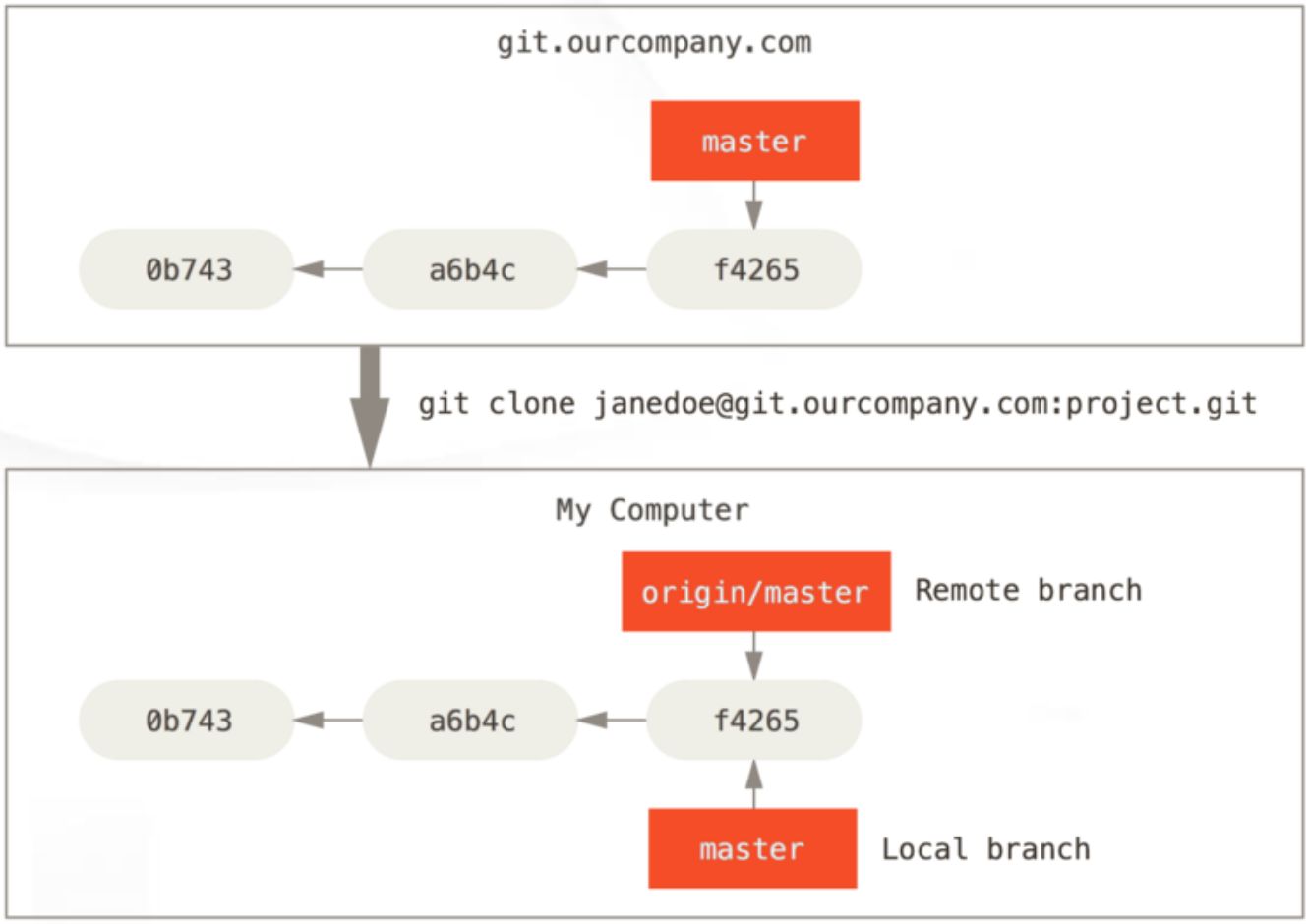
Branch Workflow



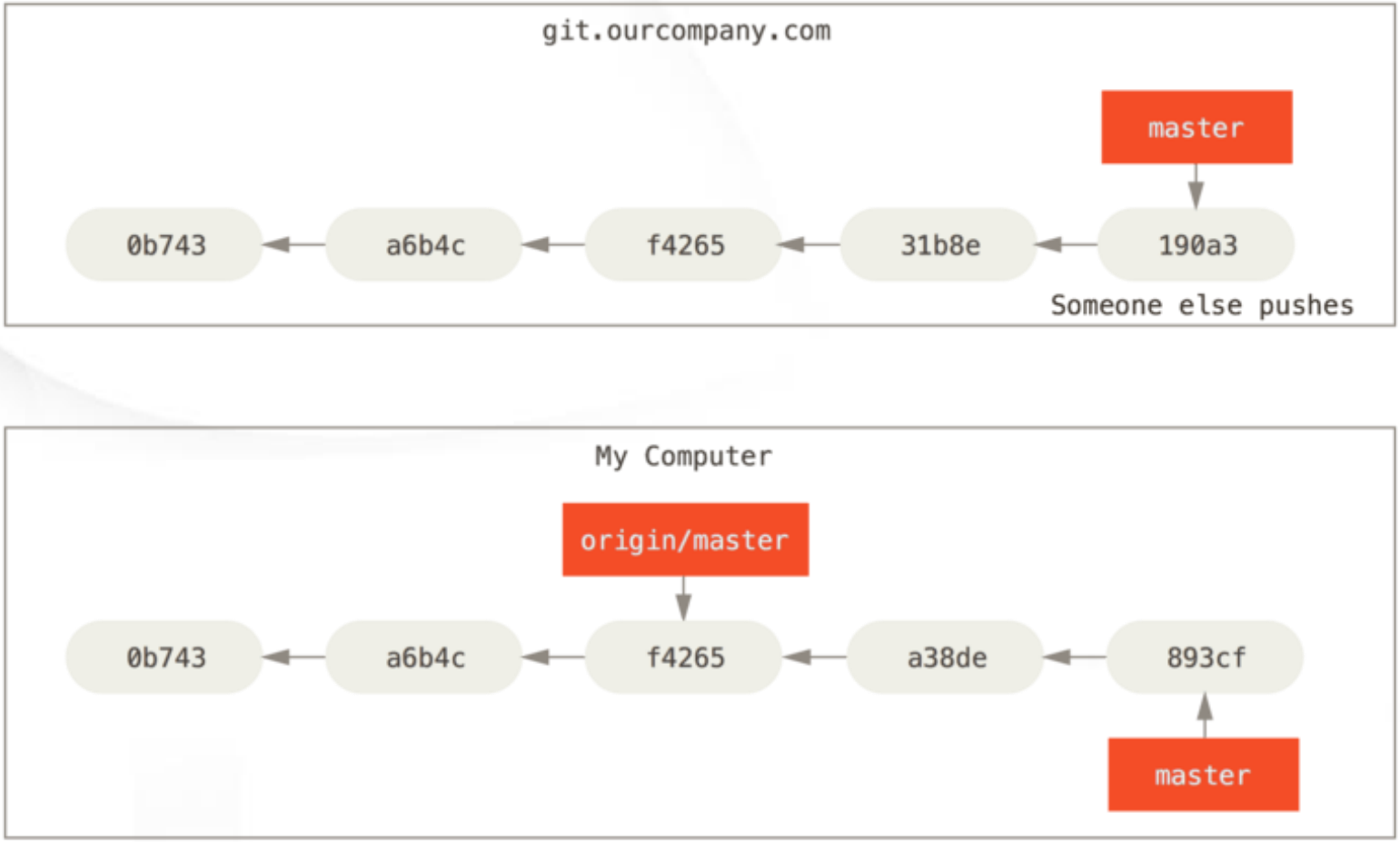
Remote Branches

Laboratório de
Tecnologia da Informação
Aplicada

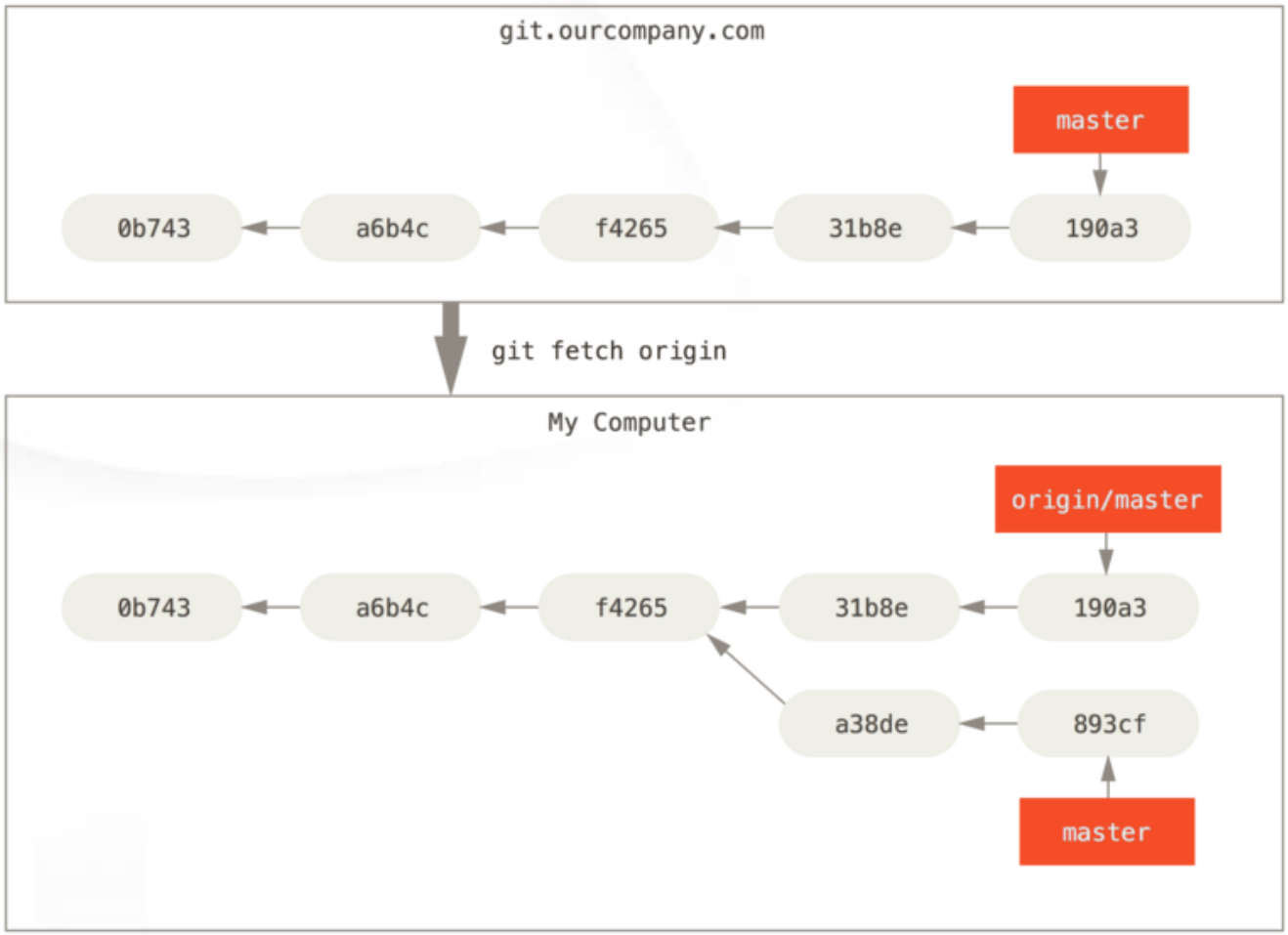
Remote Branches



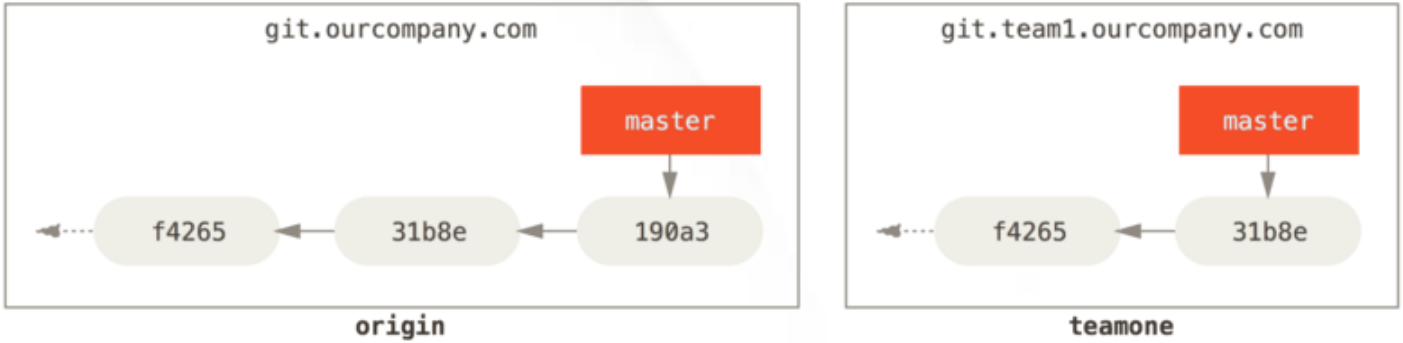
Remote Branches



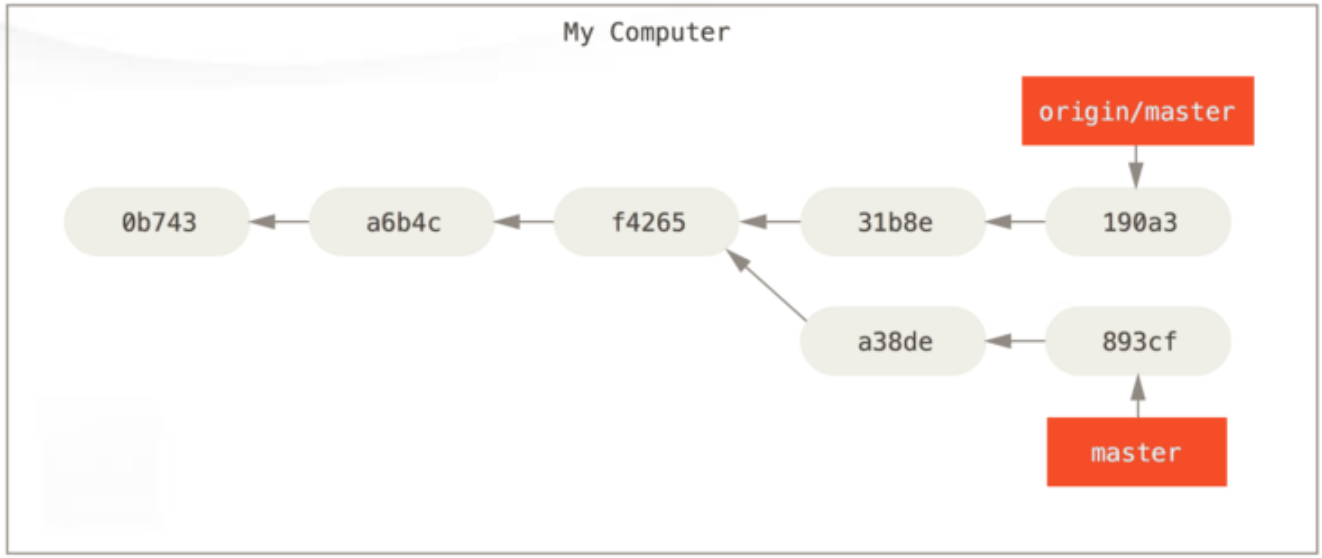
Remote Branches



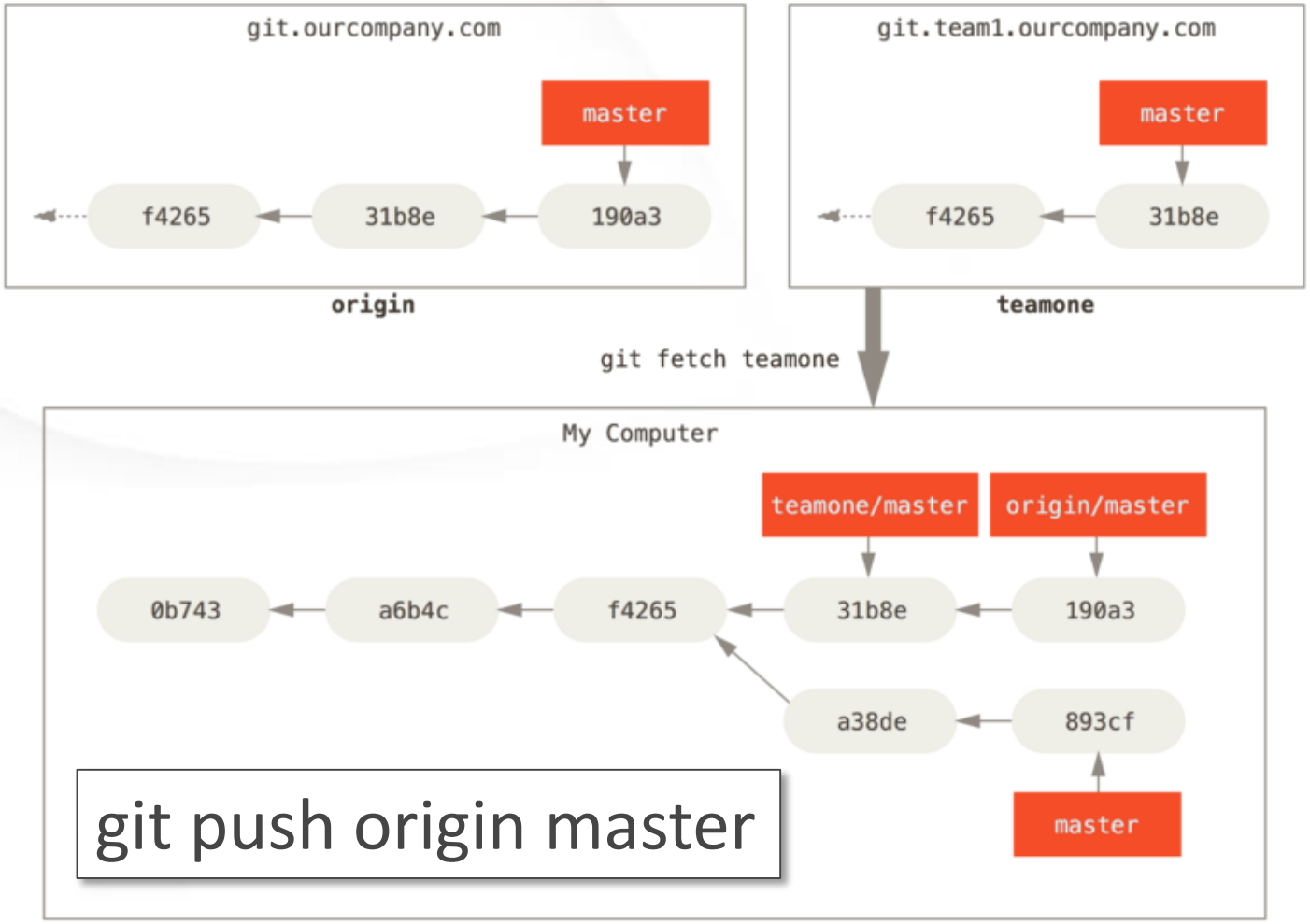
Remote Branches



git remote add teamone git://git.team1.ourcompany.com



Remote Branches



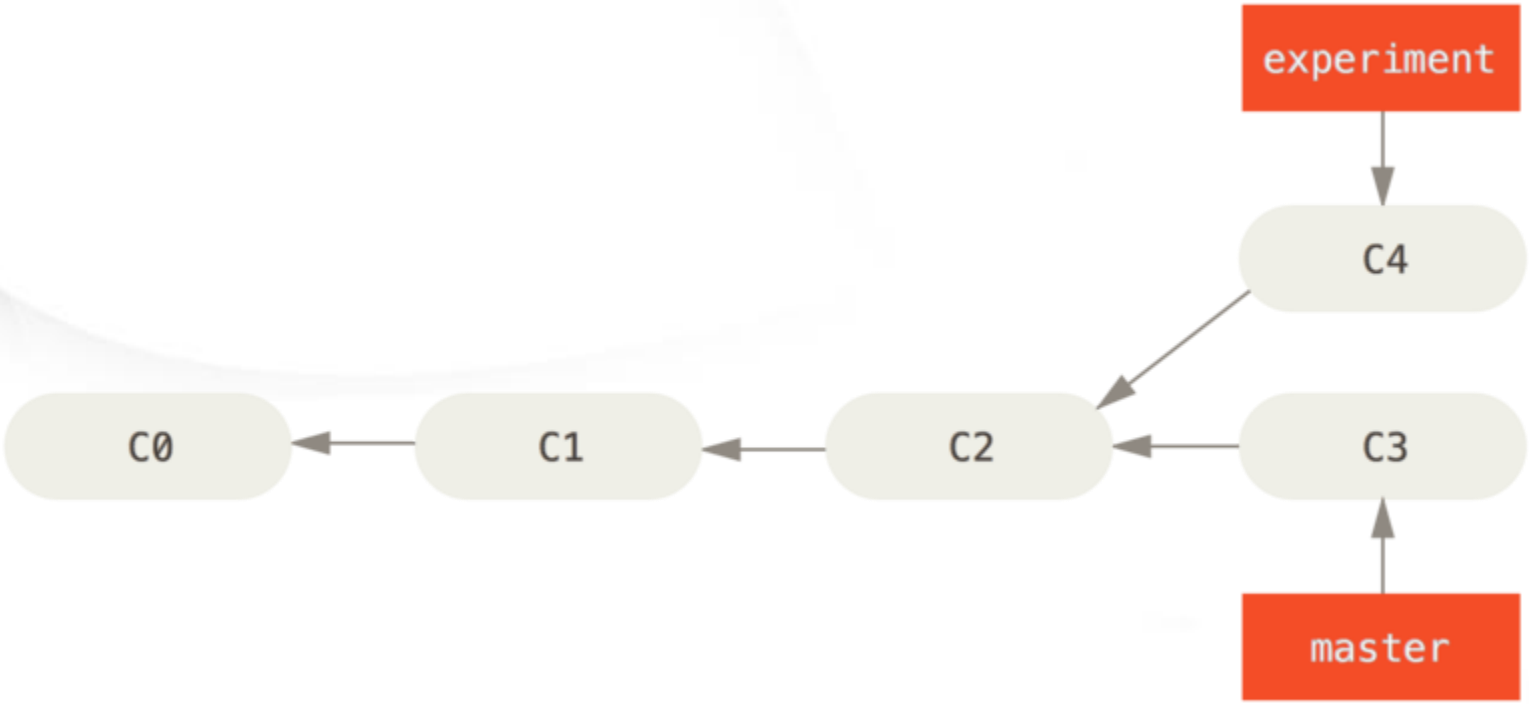
git push origin master

Credenciais

git config credential.helper store

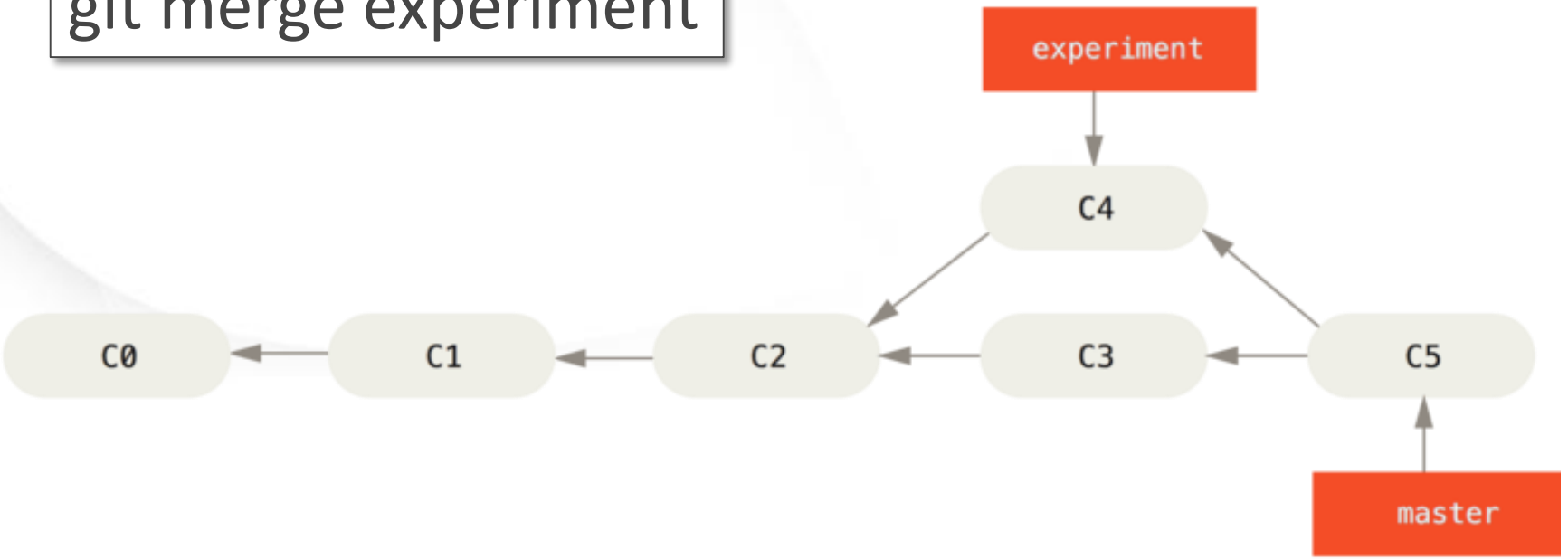
git config credential.helper wincred

Rebase



Rebase

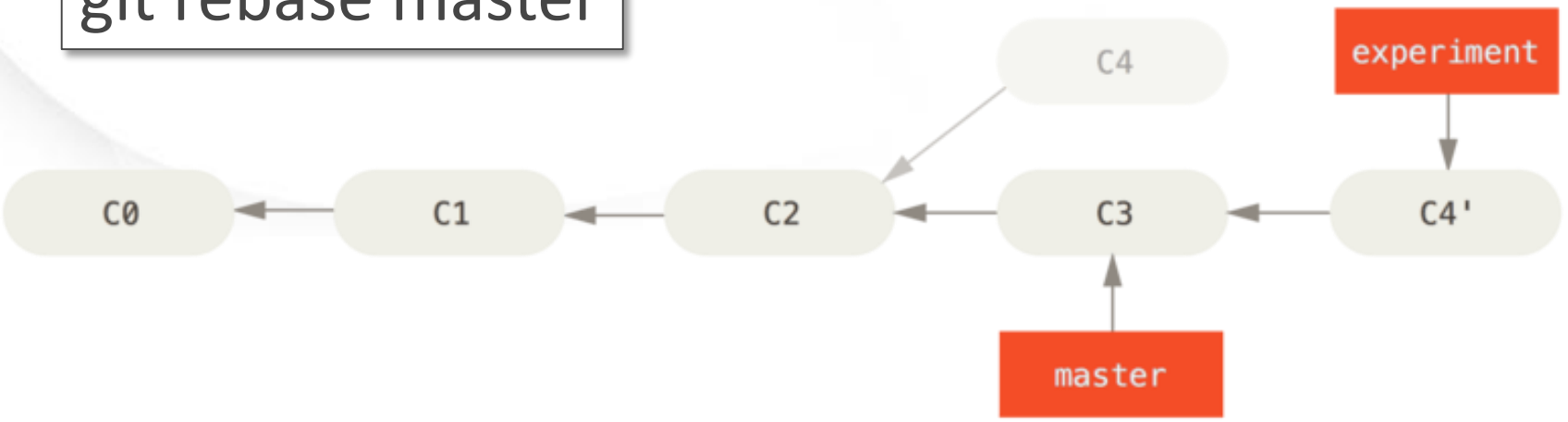
git merge experiment



Rebase

git checkout experiment

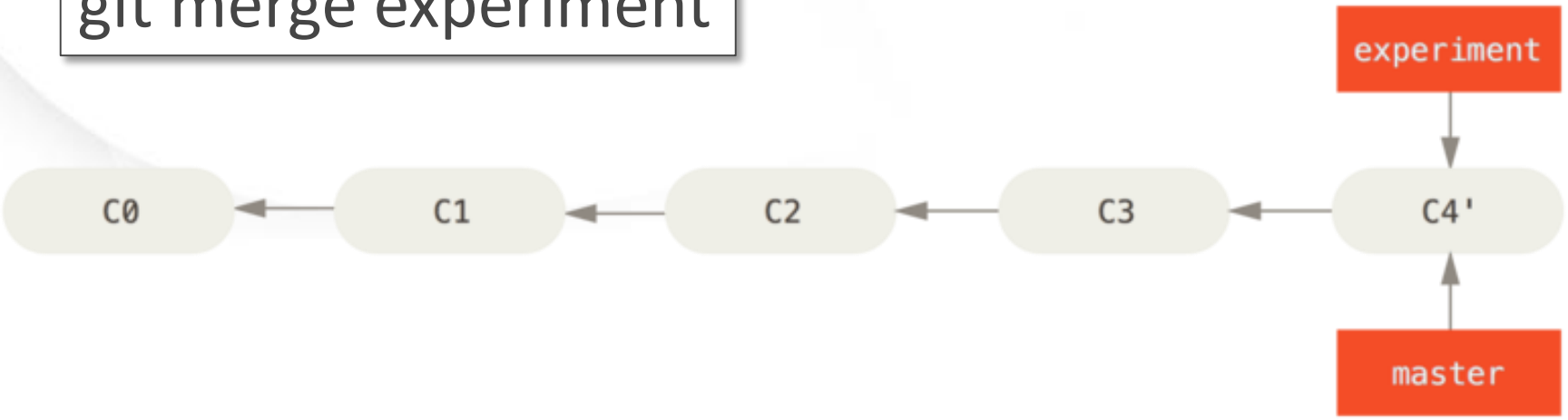
git rebase master



Rebase

git checkout master

git merge experiment



Desfazendo Mudanças

Laboratório de
Tecnologia da Informação
Aplicada

Checkout

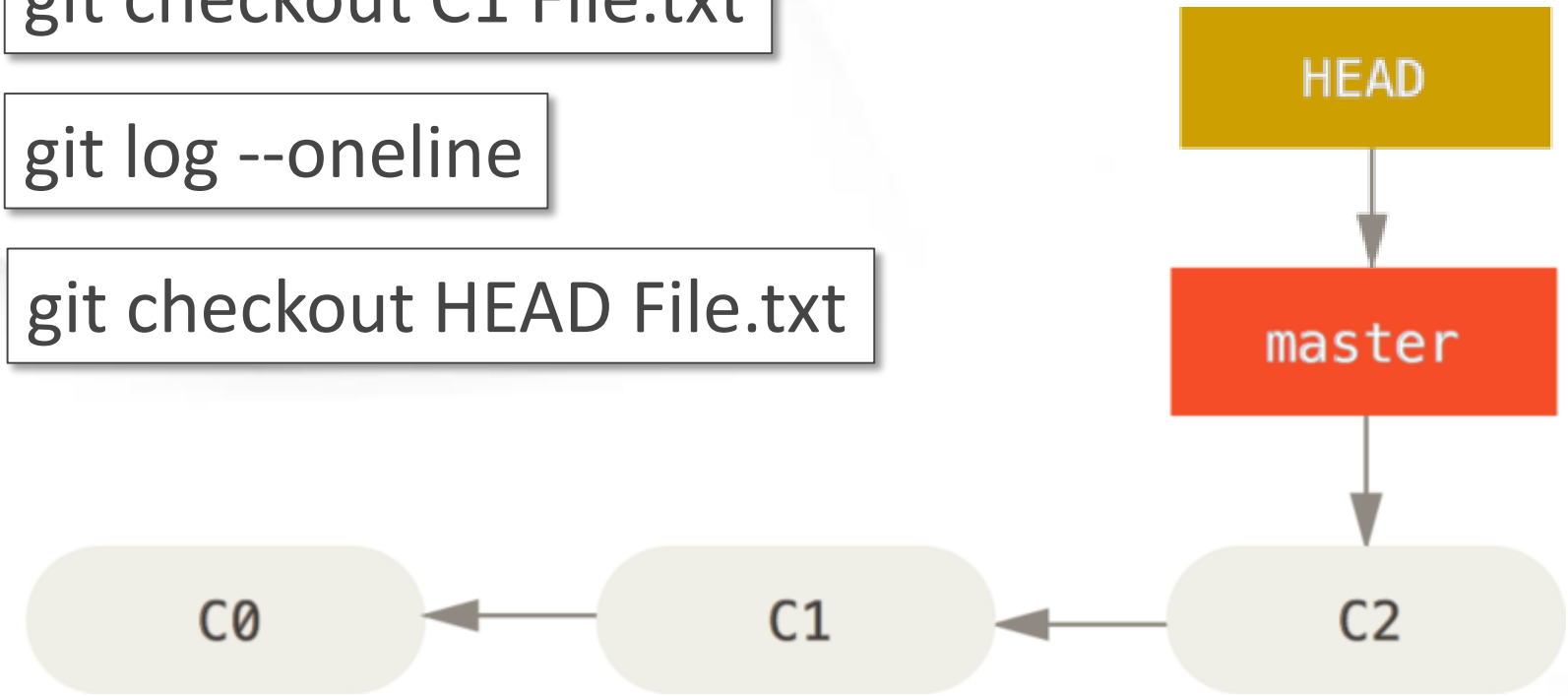
- Checkout arquivos;
- Checkout commits; ⟶ Read-Only
- Checkout branches.

Checkout - arquivo

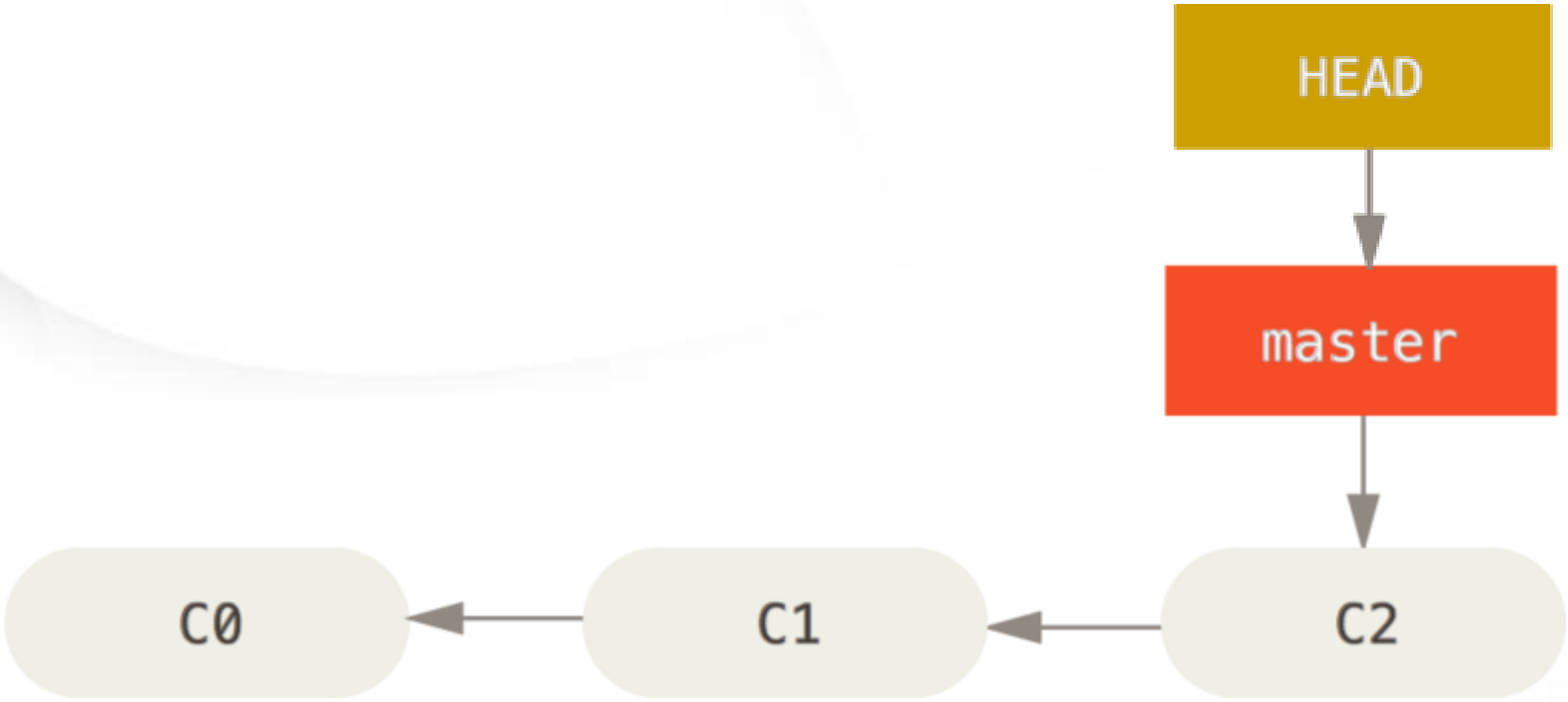
```
git checkout C1 File.txt
```

```
git log --oneline
```

```
git checkout HEAD File.txt
```

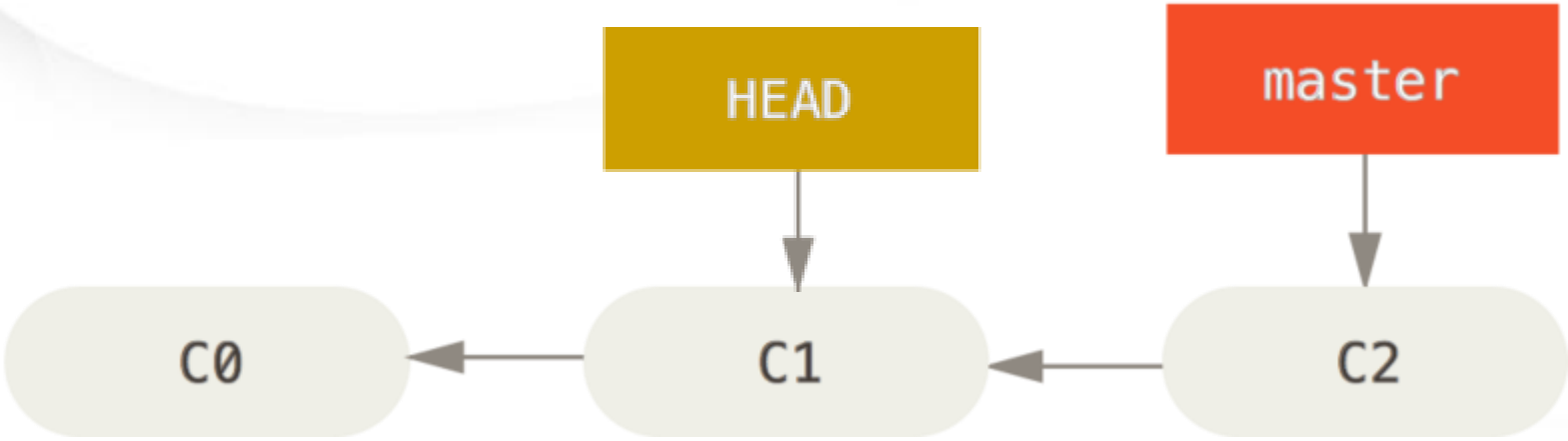


Checkout - commit



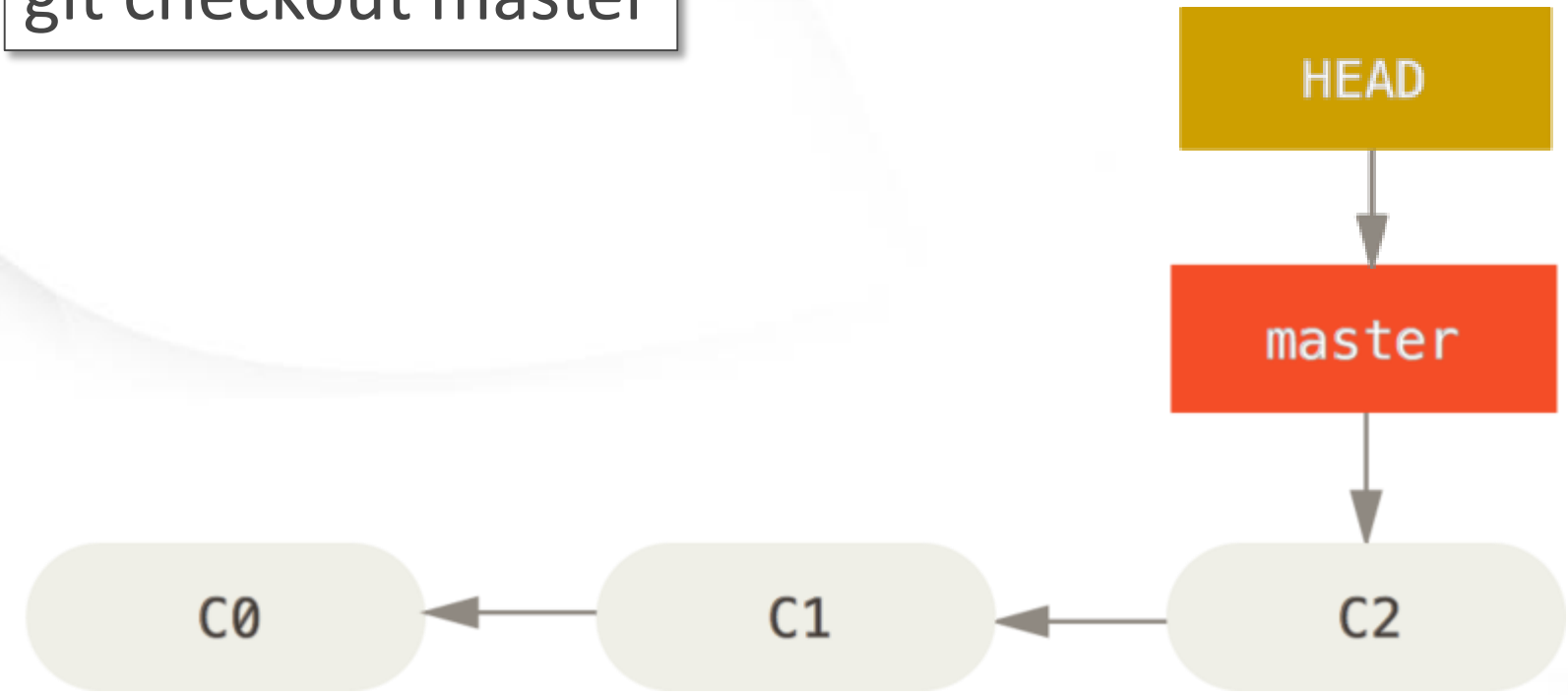
Checkout - commit

git checkout C1

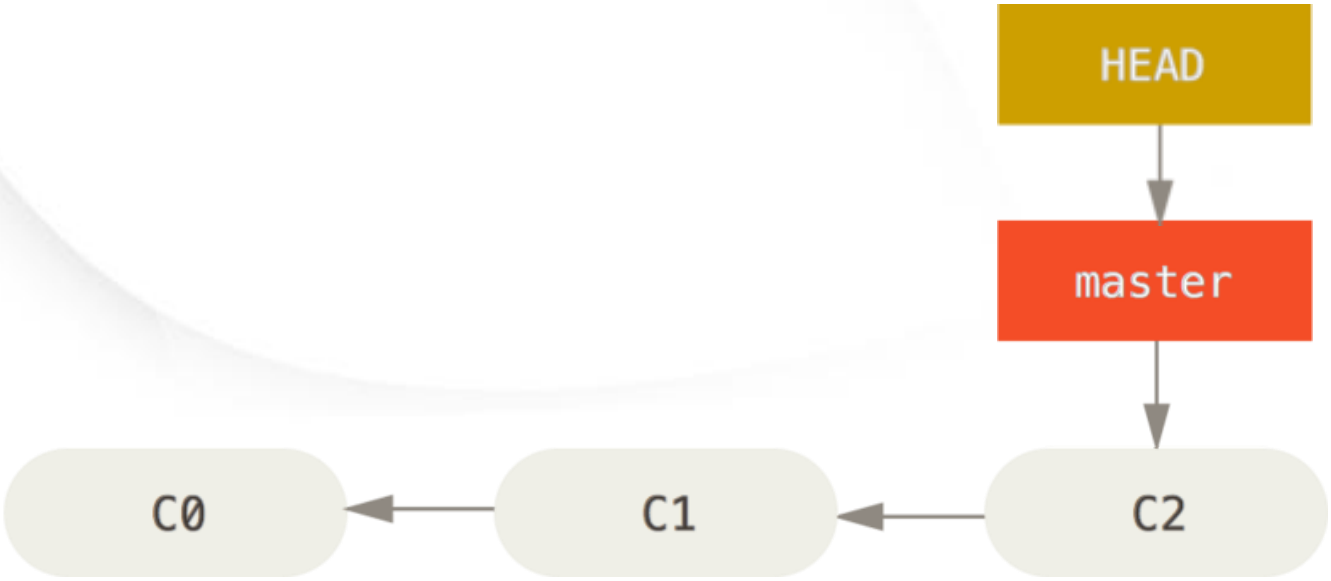


Checkout - commit

git checkout master

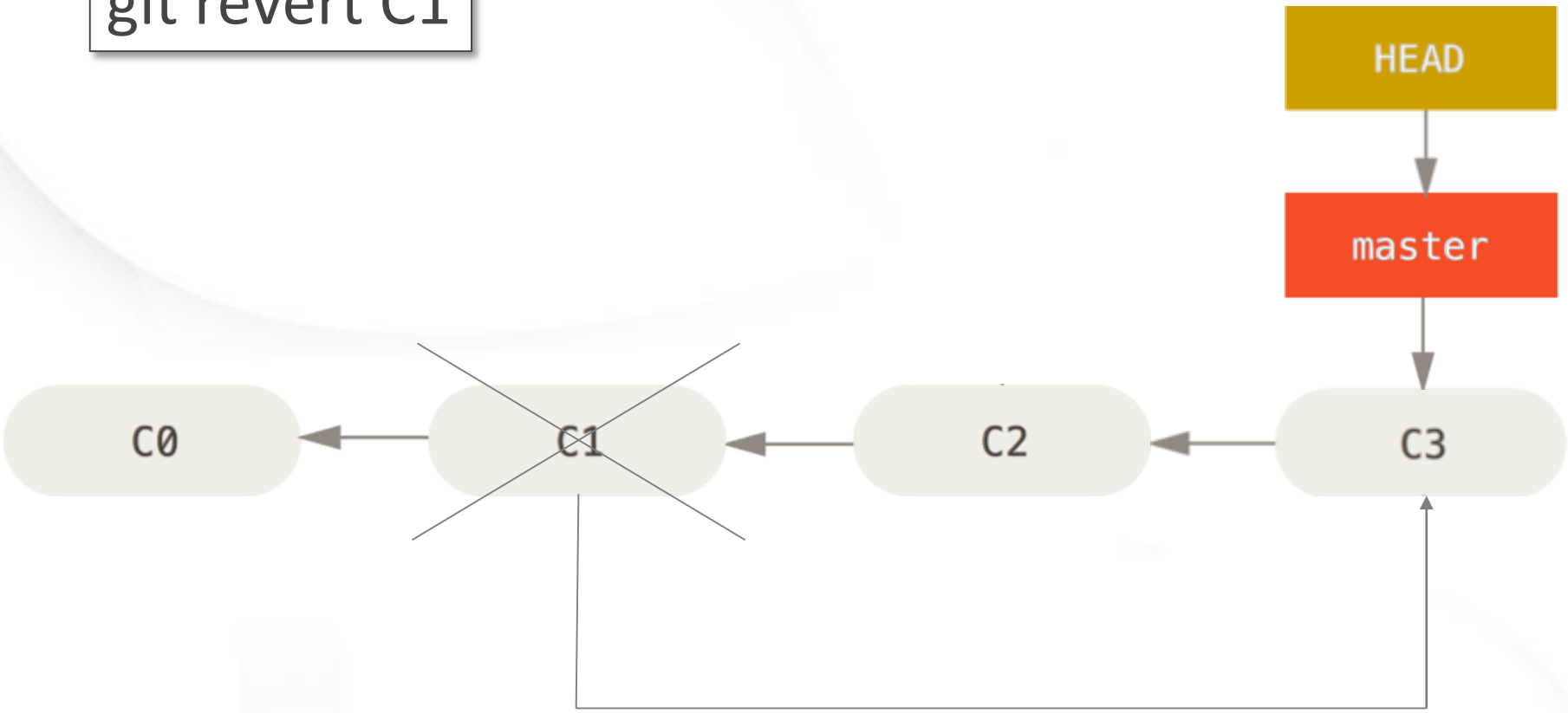


Revert



Revert

git revert C1

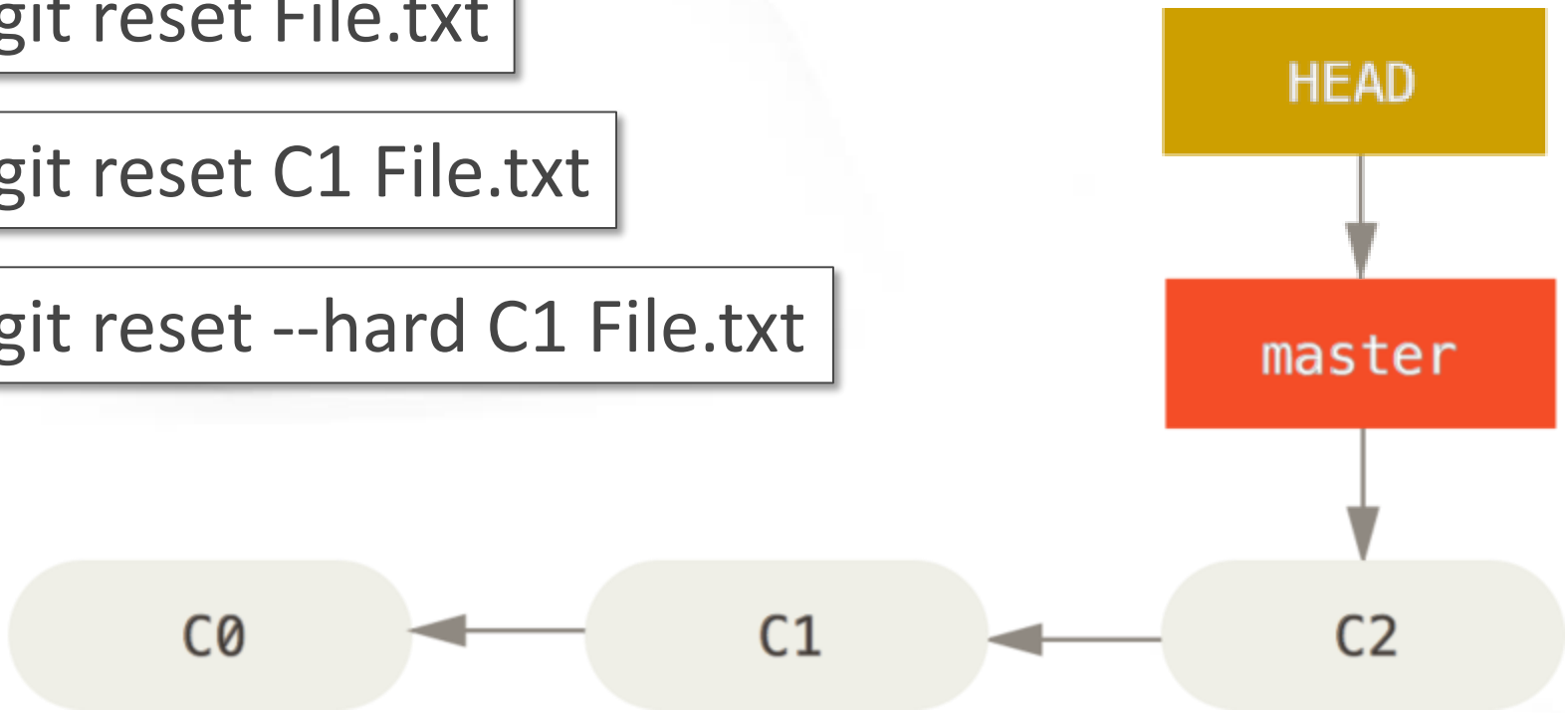


Reset - arquivo

```
git reset File.txt
```

```
git reset C1 File.txt
```

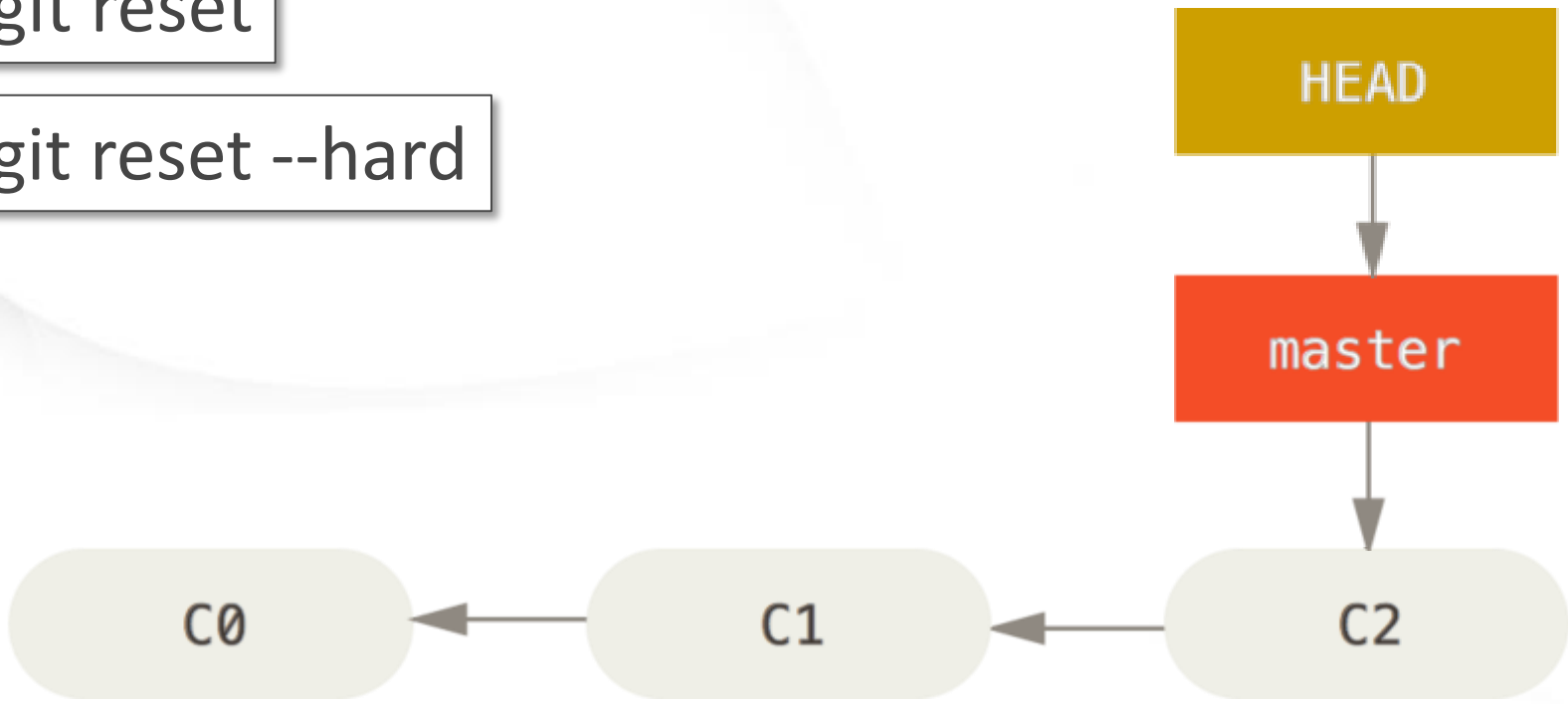
```
git reset --hard C1 File.txt
```



Reset - commit

git reset

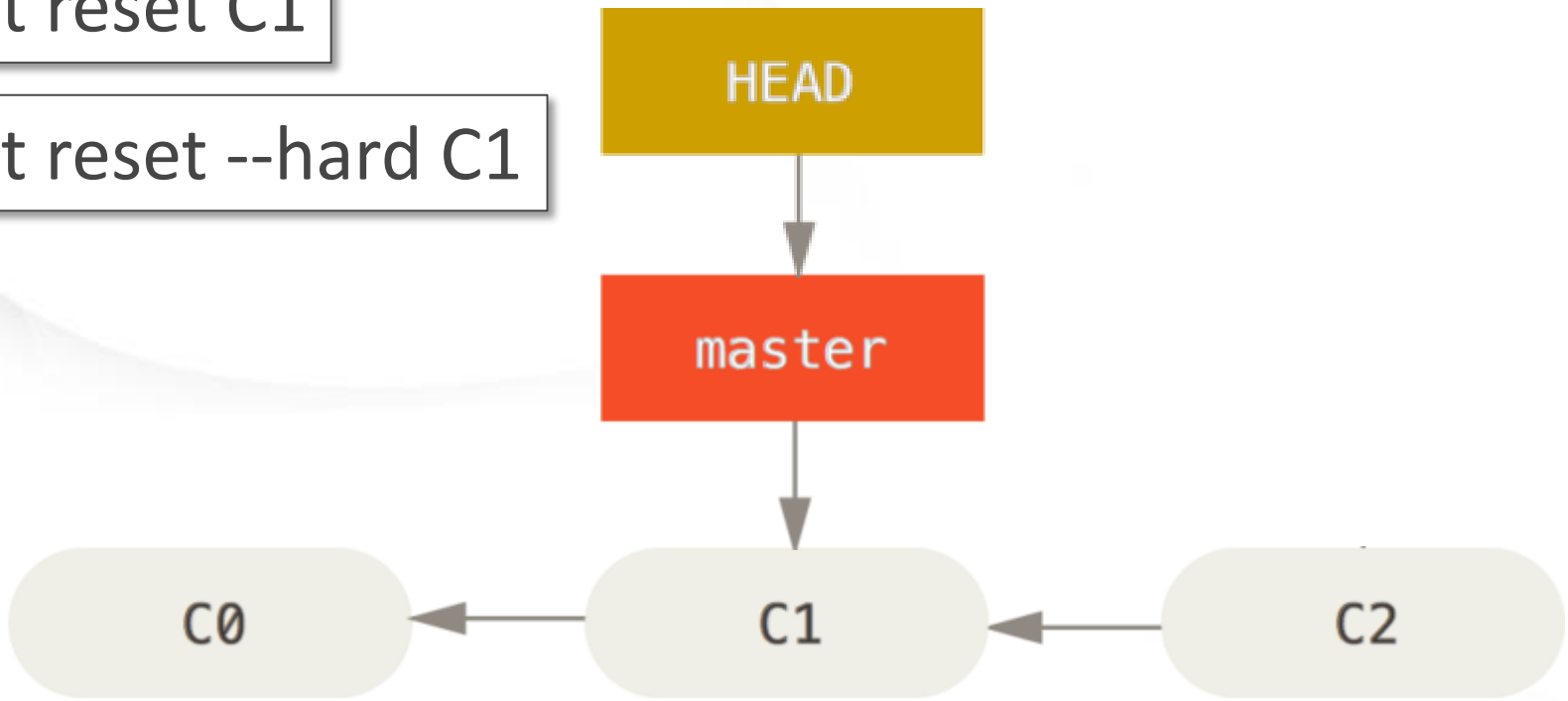
git reset --hard



Reset - commit

git reset C1

git reset --hard C1



Exercício

https://github.com/marcelocordeiro/Jornada-Unesp

git remote add upstream
https://github.com/marcelocordeiro/Jornada-Unesp

DUVIDAS????

Obrigado!

Marcelo Augusto Cordeiro

Gabriel Luiz Bastos de Oliveira

Luís Henrique Puhl de Souza

marcelo.augusto.cordeiro@gmail.com

gabiluiz@gmail.com

luispuhl@gmail.com

Laboratório de
Tecnologia da Informação
Aplicada